UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INFORMÁTICA
DOCTORADO EN CIENCIA Y TECNOLOGÍA INFORMÁTICA

TESIS DOCTORAL

Scalable Multi-Objective Optimization

Luis Martí Orosa

DIRIGIDA POR

Antonio Berlanga de Jesús

José Manuel Molina López

Colmenarejo, 2011

Web page: http://www.giaa.inf.uc3m.es/miembros/lmarti

E-mail: lmarti@inf.uc3m.es

Telephone: +34 91 856 1320

Address:

Grupo de Inteligencia Artificial Aplicada
Departamento de Informática
Universidad Carlos III de Madrid
Av. de la Universidad Carlos III, 22
Colmenarejo 28270 — Spain

Tesis Doctoral
SCALABLE MULTI-OBJECTIVE OPTIMIZATION

Autor: Luis Martí Orosa
Directores: Antonio Berlanga de Jesús
José Manuel Molina López

A mi padre y mi tío Felipe, que se fueron sin avisar.

Contents

List of Figures

List of Tables

Abstract

THIS thesis is concerned with the three open in multi-objective optimization: (i) the development of strategies for dealing with problems with many objective functions; (ii) the comprehension and solution of the model-building issues of current MOEDAs, and; (iii) the formulation of stopping criteria for multi-objective optimizers.

We argue about what elements of MOEDAs should be modified in order to achieve a substantial improvement on their performance and scalability. However, in order to supply a solid ground for that discussion, some other elements are to be discussed as well. In particular, this thesis:

- sketches the supporting theoretical corpus and the fundamentals of MOEA and MOEDA algorithms;
- analyzes the scalability issue of MOEAs from both theoretical and experimental points of view;
- discusses the possible directions of improvement for MOEAs' scalability, presenting the current trends of research;
- gives reasons of why EDAs can be used as a foundation for achieving a sizable improvement with regard to the scalability issue;
- examines the model-building issue in depth, hypothesizing on how it affects MOEDAs performance;
- proposes a novel model-building algorithm, the model-building growing neural gas (MB-GNG), which fulfills the requirements for a new approach derived from the previous debate, and;
- introduces a novel MOEDA, the multi-objective neural EDA, that is constructed using MB-GNG as foundation.

The formulation of an strategy for stopping multi-objective optimizers became obvious and necessary as this thesis was developed. The lack of an adequate stopping criterion made the rendered any experimentation that had to do with many objectives a rather cumbersome task. That is why it was compulsory to deal with this issue in order to proceed with further studies. In this regard, the thesis:

- provides an updated and exhaustive state-of-the-art of this matter;
- examines the properties and characteristics that a given stopping criterion should exhibit;

- puts forward a new stopping criterion, denominated MGBM, after the authors last names, that has a small computational footprint, and;
- experimentally validates MGBM in a set of experiments.

Theoretical discussions and algorithm proposals are experimentally contrasted with current state-of-the-art approaches when required.

Resumen

MUCHAS actividades humanas están relacionadas con la elaboración de artefactos cuyas características, organización y/o costes de producción, etc., se deben ajustar en la manera más eficiente posible. Este hecho ha creado la necesidad de tener herramientas matemáticas y computacionales capaces de tratar estos problemas, lo cual ha impulsado el desarrollo de distintas áreas de investigación interrelacionadas, como, por ejemplo, la optimización, programación matemática, investigación de operaciones, etc.

El concepto de optimización se puede formular en términos matemáticos como el proceso de buscar una o más soluciones factibles que se correspondan con los valores extremos de una o varias funciones. La mayor parte de los problemas de optimización reales implican la optimización de más de una función a la vez. Esta clase de problemas se conoce como problemas de optimización multi-objetivo (POM).

Existe una clase de POM que es particularmente atractiva debido a su complejidad inherente: los denominados *problemas de muchos objetivos*. Estos son problemas con un número relativamente elevado de funciones objetivo. Numerosos experimentos han mostrado que los métodos "tradicionales" no logran un desempeño adecuado debido a la relación intensamente exponencial entre la dimensión del conjunto objetivo y la cantidad de recursos requeridos para resolver el problema correctamente. Estos problemas tienen una naturaleza poco intuitiva y, en particular, sus soluciones son difíciles de visualizar por un tomador de decisiones humano. Sin embargo, son bastante comunes en la práctica (Stewart et al., 2008).

La optimización multi-objetivo ha recibido una importante atención por parte de la comunidad dedicada a los algoritmos evolutivos (Coello Coello et al., 2007). Sin embargo, se ha hecho patente la necesidad de buscar alternativas para poder tratar con los problemas de muchos objetivos. Los algoritmos de estimación de distribución (EDAs, por sus siglas en inglés) (Lozano et al., 2006) son buenos candidatos para esa tarea. Estos algoritmos se han presentado como una revolución en el campo de la computación evolutiva. Ellos sustituyen la aplicación de operadores inspirados en la selección natural por la síntesis de un modelo estadístico. Este modelo es muestreado para generar nuevos elementos y así proseguir con la búsqueda de soluciones. Sin embargo, los EDAs multi-objetivo (MOEDAs) no han logrado cumplir las expectativas creadas a priori.

El leit motif de esta tesis se puede resumir en que la causa principal del bajo rendimiento MOEDAs se debe a los algoritmos de aprendizaje automático que se aplican en la construcción de modelos estadísticos. Los trabajos existentes hasta el momento han tomado una aproximación de "caja negra" al problema de la construcción de modelos. Por esa razón, se aplican métodos de aprendizaje automático ya existentes sin modificación alguna, sin percatarse que el problema de la construcción de modelos para EDAs tiene unos requisitos propios que en varios casos son contradictorios con el contexto original de aplicación de los mencionados algoritmos. En particular, hay propiedades compartidas por la mayoría de los enfoques de aprendizaje automático que podrían evitar la obtención de una mejora sustancial en el resultado de los MOEDAs. Ellas son:

- el tratamiento incorrecto de los valores atípicos (outliers) en el conjunto de datos;
- tendencia a la pérdida de la diversidad de la población, y;
- exceso de esfuerzo computacional dedicado a la búsqueda de un modelo óptimo.

Estos problemas, aunque ya están presentes en los EDAs de un solo objetivo, se hacen patentes al escalar a problemas de varios objetivos y, en particular, a muchos objetivos. Además, con el aumento de la cantidad de objetivos con frecuencia esta situación se ve agravada por las consecuencias de la "maldición de la dimensionalidad".

La cuestión de los valores atípicos en los datos es un buen ejemplo de como la comunidad no ha notado esta diferencia. En el contexto tradicional del aprendizaje automático los valores extremos son considerados como datos ruidosos o irrelevantes y, por tanto, deben ser evitados. Sin embargo, los valores atípicos en los datos de la construcción de modelos representan las regiones recién descubiertas o soluciones candidatas del conjunto de decisión y por lo tanto deben ser explorados. En este caso, los casos aislados debe ser al menos igualmente representados por el modelo con respecto a los que están formando grupos.

Sobre la base de estos razonamientos se estructuran los principales resultados obtenidos en el desarrollo de la tesis. A continuación se enumeran brevemente los mismos mencionando las referencias principales de los mismos.

- *Comprensión del problema de la construcción de modelos en MOEDAs (Martí et al., 2010a, 2008b, 2009c).* Se analiza que los EDAs han asumido incorrectamente que la construcción de modelos es un problema tradicional de aprendizaje automático. En el trabajo se muestra experimentalmente la hipótesis.
- *Growing Neural Gas: una alternativa viable para construcción de modelos (Martí et al., 2008c).* Se propone el Model-Building Growing Neural Gas network (MB-GNG), una modificación de las redes neuronales tipo Growing Neural Gas. MB-GNG tiene las propiedades requeridas para tratar correctamente la construcción de modelos.
- *MONEDA: mejorando el desempeño de los MOEDAs (Martí et al., 2008a, 2009b, 2010c).* El Multi-objective Optimization Neural EDA (MONEDA) fue ideado con el fin de hacer frente a los problemas arriba descritos de los MOEDAs y, por lo tanto, mejorar la escalabilidad de los MOEDAs. MONEDA utiliza MB-GNG para la construcción de modelos. Gracias a su algoritmo específico de construcción de modelos, la preservación de las élites de individuos de la población y su mecanismo de sustitución de individuos MONEDA es escalable capaz de resolver POMs continuos de muchos objetivos con un mejor desepeño que algoritmos similares a un coste computacional menor. Esta propuesta fue nominada a mejor trabajo en GECCO'2008.
- *MONEDA en problemas de alta complejidad (Martí et al., 2009d).* En este caso se lleva a cabo una amplia experimentación para comprender como las características de MONEDA provocan una mejora en el desempeño del algoritmo, y si sus resultados mejoran los obtenidos de otros enfoques. Se tratan problemas de alta complejidad. Estos experimentos demostraron que MONEDA produce resultados sustancialmente mejores que los algoritmos similares a una menor coste computacional.

- *Nuevos paradigmas de aprendizaje: MARTEDA (Martí et al., 2010d)*. Si bien MB-GNG y MONEDA mostraron que la vía del tratamiento correcto de la construcción de modelos era una de las formas de obtener mejores resultados, ellos no evadían por completo el punto esencial: el paradigma de aprendizaje empleado. Al combinar un paradigma de aprendizaje automático alternativo, en particular, la Teoría de Resonancia Adaptativa, se trata a este asunto desde su raíz. En este respecto se han obtenido algunos resultados preliminares alentadores.

- *Criterios de parada y convergencia (Martí et al., 2007, 2009a, 2010e)*. Con la realización de los experimentos anteriores nos percatamos de la falta de de un criterio de parada adecuado y que esta es un área inexplorada en el ámbito de la investigación en algoritmos evolutivos multi-objectivo. Abordamos esta cuestión proponiendo una serie de criterios de parada que se han demostrado efectivos en problemas sintéticos y del mundo real.

Agradecimientos

Sin dudas muchas personas han influido en el desarrollo de mi vida profesional y como consecuencia de ello, en esta tesis.

Ante todo, debo que agradecer a mis padres, Teresa y Luis, y al resto de mi familia. Ellos, además de ser la razón por la que me encuentro en el mundo, lo cual es ya motivo suficiente, me supieron inocular desde pequeño la pasión por la investigación y la academia.

Esta tesis es resultado de la guía, paciencia, apoyo y consejos de mis directores de tesis, Antonio Berlanga y José Manuel Molina. Sus puertas siempre estuvieron abiertas para mis preocupaciones, ocurrencias y problemas. He aprendido de ellos y con ellos. Muchas gracias.

Quiero agradecer de manera especial a Luciano García y a Alberto Policriti, con los que comencé mi carrera investigativa, y a Carlos A. Coello Coello, que me recibió durante mi estancia en el CINVESTAV con los brazos abiertos.

He compartido la singladura de este doctorado con mis colegas de la universidad y les tengo que dar gracias por ello. Ellos, además de compañeros, han sabido ser amigos. Muchas gracias a Jesús que se implicó en nuestros artículos de manera especial. Mi aprecio especial a Álvaro, Elisenda, Gonzalo, Heike, Javier, José Daniel, José Luis, Juan, Juanita, Kike, Mar, Miguel Ángel, Mirren, Roberto, Rodrigo y Verónica. Muchas gracias por vuestra ayuda, coloquios y cafés.

He compartido con Nayat, además, la experiencia de vivir lejos de casa y en ambas cosas nos hemos sabido acompañar. Mis dos compañeros de despacho, Óscar y Federico, han tolerado estoicamente mis inquietudes y tertulias. Igualmente, Ramón, junto a su inefable amistad, me brindó su paciencia mientras le perseguía por los pasillos preguntándole sobre estadística. Agradecimientos aparte se merece mi inseparable compañero de aventuras y desventuras castellano-manchegas-colmenarejanas, César de Pablo, y su a su vez inseparable Isabel, por todo lo que han sabido compartir.

Fuera de la universidad, muchos amigos han sabido estar presentes, a veces a pesar de la distancia, según hacía esta tesis. Quiero agradecerles sus ánimos, el aprender que "¿. . . y el doctorado, como va?" era una pregunta tabú y por saber disculpar por mis ausencias. Particularmente a Abel, Alexei, Alicia, Beder, Camilo, Ernesto, Damián, Daria, Marco, Milena, Sole y Yadira. Sé que seguramente me dejo a alguno fuera y le pido disculpas de antemano.

Algunos de estos amigos, además de brindarme su amistad, me han ayudado a corregir este texto. Gracias a ellos, este documento tiene una calidad superior. Sin embargo, debo decir que cualquier error en el texto es de responsabilidad mía.

Luis Martí Orosa
Colmenarejo, marzo 2011.

1

Introduction

You know that I write slowly.

Karl Friedrich Gauss

MOST human endeavours involve the creation of artifacts whose properties, building costs, and schedules must be tuned to be as efficient as possible. This fact has prompted the creation of a number of interrelated research areas like optimization, mathematical programming, operational research and decision-making. Although these areas share some of their goals, each of them differs from the others in the approaches put forward by their respective communities and the characteristics of the problems they deal with.

The optimization concept can be formulated in mathematical terms as the process of searching for one or more feasible values that corresponds to the extreme values of one or more functions.

The interest raised by optimization is not limited to its direct application scope. Many other classes of problems, like pattern recognition, prediction or clustering, can be posed as optimization problems where a given error function must be minimized.

Many theoretic and real-life problems are, or can be posed as, optimization problems. Frequently, these problems involve more than one aspect to be optimized, since besides the need of improving a certain feature, this improvement must be balanced with its cost, its production time, its robustness, etc. This class of problems is known as *multi-objective optimization problems* (MOPs). In MOPs there is a set of functions whose values must be optimized. Therefore, an optimizer's solution is a set of trade-off, equally good solutions.

Even a simple situation, like deciding what fruit to eat, calls for the analysis of different factors. Figure 1.1 epitomizes this problem. Different fruits have different tastes, but some tasty fruits, like pineapples, require a rather cumbersome preparation process. Similarly, according to the figure, other fruits that might not be so tempting in return are relatively simple to eat. That is the case of the apples in our example. In these cases, we would sacrifice flavor in favor of a kind of "user friendliness". Both classes of food might be incomparable, as they represent different trade-offs that a decision maker at a higher level must evaluate.

Still, it is obvious that peaches, strawberries and seedless grapes offer better combinations of flavor and difficulty, while it becomes apparent that grapefruit is probably the worst of all fruits, as it lacks both properties. Many other decision targets can be included, like vitamin content, transportation requirements, etc. As more objectives are added it the complexity

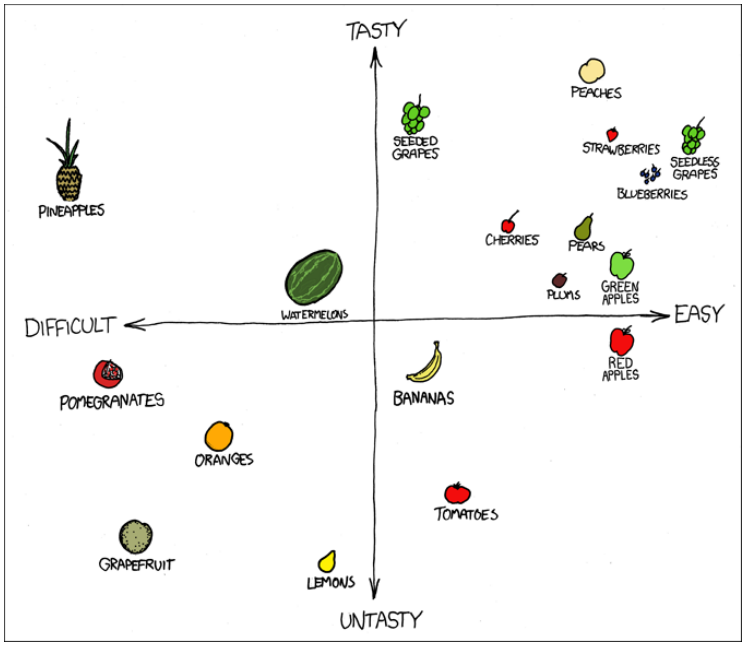


Figure 1.1: An example of a common-life multi-objective problem: choosing which fruit to eat. In this case two objectives are taken into account in order to make a decision, the tastiness of the fruit and how difficult is that fruit to eat. Comic by Randall Munroe (xkcd), `http://xkcd.org/388/`; used with permission.

of the analysis escalates, making the problem harder to understand and solve. To further complicate this problem, restrictions like allergies or maximum calories intake might be also required to be taken into account.

1.1 Multi-Objective Optimization

The determination of the better trade-offs is the purpose of multi-objective optimization. The concept of multi-objective optimization refers to the process of finding one or more feasible solutions to a problem by trading off the equally optimal values of two or more functions subject to a set of constraints. Formal approaches to this subject were pioneered by Edgeworth (1881), (Auspitz and Lieben, 1889) and Pareto (1896).

MOPs have been address with a variety of methods (Branke et al., 2008; Ehrgott, 2005; Miettinen, 1999). Approaches to these problems can be grouped into two broad groups: directed search methods and population-based methods.

The development of multi-objective optimization methods is rooted in “classical” (single-objective) optimization, with the first concerns with practical application flourishing as part of the efforts dedicated to operations research during and after Second World War.

The main issue regarding the extrapolation of numerical optimization tools (Nocedal and Wright, 1999) to the multi-objective domain is that they require gradient information in order to determine a search direction. These methods update the position of a search point by following the information yielded by first and second order derivatives.

This poses two problems. First, that the objective functions must meet some mathematical restrictions, like continuity and differentiability, which are rather hard to satisfy in real-world problems. Second, and perhaps more important, that all objectives most be taken into account at the same time.

This second matter is probably the cornerstone issue of multi-objective research topics. The straightforward approach is to combine all objectives into an aggregate function and, therefore converting a multi-objective problem in a single-objective one

A wide variety of scalarization alternatives exist (Eichfelder, 2008). However, these methods have the inherent drawback derived from any dimensionality reduction, that necessarily implies losing information. After transforming a MOP into a scalarized form multiple, "traditional" optimization methods can be used. Depending on the nature of the problem, the characteristics of the decision and objective spaces, etc. different methods can be applied. For example, multi-criteria linear programming, simplex, steepest descent, conjugate-gradient methods, tabu search, etc.

The application of these methods has another implicit disadvantage: they yield only one solution. Therefore, it is not straightforward how to obtain a good sampling of the different trade-offs.

If a MOP has certain characteristics, e. g., linearity or convexity of the objective functions, the solution can be exactly determined by mathematical programming approaches (Branke et al., 2008). However, in the general case, finding the solution of this class of problems is an *NP*–complete problem (Bäck, 1996). In this case, heuristic or metaheuristic methods can be applied in order to have solutions of practical value at an admissible computational cost.

Population-based heuristics are one of the ways of overcoming this issue. These approaches have a stochastic nature. They explore different zones of the decision set in parallel. Although in most cases they sacrifice the theoretical robustness and convergence properties in return they are capable of yielding usable solutions at a reasonable computational cost. Evolutionary algorithms (EAs) (Bäck, 1996; Bäck et al., 1997) have proved themselves as a valid and competent approach from theoretical and practical points of view to these issues.

1.2 Evolutionary Approaches to Multi-Objective Optimization

Scientists and engineers have frequently resorted to nature to look for assistance and inspiration when solving problems. We can find evidences of this in almost every corner of human creative activity like, for example, aircraft design, architecture, pharmacology, etc. Therefore, it is not surprising that computer scientists have also turned to nature when seeking solutions to their problems.

This has led to the development of a set of closely related research areas known as soft computing, computational intelligence or nature-inspired computing. This area encompasses different techniques that are somewhat based on structures and processes existing in nature. For example, neural networks are machine-learning methods that drew inspiration from the nerve system of higher animals, fuzzy logic is based on the imprecise and vague articulation of human language and common-life reasoning, etc.

Evolutionary computation (EC) (Bäck et al., 1997; Fogel, 2006) is another example of a nature-inspired computational approach. *Evolutionary algorithms* (EAs) are optimization methods based on the gradual improvement of species that takes place in evolutionary processes.

EAs were initially applied to single-objective problems, but since the last 20 years they tackling of multi-objective problems have gained a lot of attention in their research community. This interest can be attributed in part to the large demand for practical applications but also its appealing theoretical aspects.

The *evolutionary algorithm* concept is a rather generic term used to refer to a population-based metaheuristic optimization algorithm that uses mechanisms inspired by biological theory of evolution (Darwin, 1859) and, in some cases, also influenced by other nature-related notions like Lamarckian inheritance (Lamarck, 1809), the different variants of swarm intelligence (Engelbrecht, 2006), memes theory (Dawkins, 1976; Hofstadter, 1985), among others.

An evolutionary algorithm maintains a population of individuals. Each individual represents a candidate solution of the problem being solved as its *chromosome*. These chromosomes store the information of a point of the search space in a direct or indirect representation. They can be viewed as a vector of features that can be evaluated as a candidate solution to the problem.

A *fitness assignment function* determines how fit or apt is an individual to the environment within it lies; or, in other words, how good is the solution represented by the individual with respect to the rest of the population. The individuals of the population are recombined and improved using *evolutionary operators* inspired by the reproduction, crossover and mutation processes of nature. Individuals with better fitness values are more likely to take part of these processes in order to promote the gradual improvement of the population by generating progressively better individuals.

The application of EAs to MOPs has prompted the creation of what has been called *multi-objective optimization evolutionary algorithms* (MOEAs) (Coello Coello et al., 2007; Deb, 2001). MOEAs and their related issues have become one of the hottest topics in EC research, receiving the attention of 38.8% and 45.9% of academic and industry researchers, respectively, in the year 2007 (Hornby and Yu, 2007).

Their success is due to the fact that EAs do not make any assumptions about the underlying fitness landscape. Therefore, it is believed they perform consistently well across all types of problems, although it has been shown that they share theoretical limits imposed by the no-free-lunch theorem (Corne and Knowles, 2003). Another important benefit arises from the parallel search. Thanks to that, the algorithm can produce a set of equally optimal solutions instead of one, as many other algorithms do.

When extrapolating EAs to the multi-objective domain a fundamental issue emerges: how to handle the multiple objectives functions? In this case, the fitness assignment should be capable of represent each individual fitness' by combining the different values of the objectives into a composite scalar indicator. Perhaps this matter and its consequences is the corner-stone issue in MOEA research.

1.3 Many-Objective Problems

Although MOEAs have successfully solved many complex synthetic and real-life MOPs, the majority of works has been concentrated on low dimensional problems (Deb and Saxena, 2005). MOPs dimensionality regards to two quantities:

1. the number of variables that take part of the problem, and;
2. the number of objective functions to be optimized.

Albeit the increase in the amount of variables has a direct impact of the computational cost of evaluating the functions, the addition of more functions is an issue much harder to deal with (Knowles and Corne, 2007). One topic that remains not properly dealt with inside the MOEA community is the scalability of the MOEAs when facing these problems (Coello Coello, 2006a,b) or what has been denominated as the *many-objective problems* (Purshouse, 2003; Purshouse and Fleming, 2003).

This type of problems, although counterintuitive and hard to visualize for a human decision maker, are not uncommon in real-life engineering practice like, for example, in aircraft design (Brandte and Malinchik, 2004), land use planning (Stewart et al., 2004), optimization of trackers for air traffic management and surveillance systems (Besada et al., 2005; García et al., 2009), bridge design (Nakayama et al., 1995), optical lens design, etc. (see Stewart et al. (2008) for a survey on these problems).

As more objective functions are added, the optimization algorithms suffer heavily under the curse of dimensionality (Bellman, 1961); requiring an exponential increase of the resources made available to them (see Khare et al. (2003); Praditwong and Yao (2007); Purshouse and Fleming (2007) and Deb (2001, pp. 414–419)).

There has been some studies directed towards reducing of the number of objective functions to a minimum (Brockhoff et al., 2008; Brockhoff and Zitzler, 2007a; Deb and Saxena, 2005, 2006) and, therefore, mitigating the complexity of a given problem. Although these works provide a most useful tool for alleviating the burden of a given problem they do not ultimately address the essential issue. Instead, they just postpone it.

Another viable approach is to employ cutting-edge evolutionary algorithms that would deal with high-dimensional problems more efficiently. The incorporation of learning as part of the search processes has been nominated as a viable solution (Corne, 2008). One of the forms of including learning as part of the search process is to apply estimation of distribution algorithms.

1.4 Estimation of Distribution Algorithms

One of the forms of carrying out this integration is the application of *estimation of distribution algorithms* (EDAs) (Baluja, 1994; Larrañaga and Lozano, 2002; Lozano et al., 2006; Mühlenbein and Paaß, 1996; Pelikan et al., 2006a).

EDAs have been claimed as a paradigm shift in the field of evolutionary computation. Like EAs, EDAs are population based optimization algorithms. However, in EDAs the step

where the evolutionary operators are applied to the population is substituted by the construction of a statistical model of the most promising subset of the population. This model is then sampled to produce new individuals that are merged with the original population following a given substitution policy. Because of this model-building feature EDAs have also been called *probabilistic-model-building genetic algorithms* (PMBGAs) (Pelikan et al., 1999b). A framework similar to EDAs is proposed by the *iterated density estimation evolutionary algorithms* (IDEAs) (Bosman, 2003).

The introduction of machine learning techniques implies that these new algorithms lose the biological plausibility of its predecessors. In spite of this, they gain the capacity of scalably solve many challenging problems, often significantly outperforming standard EAs and other optimization techniques.

The extension of EDAs to the multi-objective domain has lead to what can be denominated *multi-objective optimization EDAs* (MOEDAs). However, most MOEDAs have limited themselves to port single objective EDAs to the multi-objective domain by incorporating some features taken from MOEAs. Although MOEDAs have proved themselves as a valid approach to the MOP, this later fact hinders the achievement of a significant improvement regarding "standard" MOEAs.

An analysis of the results yielded by current multi-objective EDAs and their scalability against the number of objectives leads to the identification of some issues that could be preventing MOEDAs from getting substantially better results than other evolutionary approaches. Such issues include:

1. incorrect treatment of isolated individuals;
2. loss of population diversity; and
3. excess of computational effort devoted to finding an optimal population model.

These issues can be traced back to the single-objective predecessor of most MOEDAs and its respective model-building algorithms. The cause of these behaviors may be attributed to the fact that those methods are not meant specifically for the problem we are dealing with here. These behaviors, although justified in the original field of application of the algorithms, might hinder the performance of the process, both in the accuracy and in the resource consumption senses.

In the statistical and machine learning areas the data instances that are relatively isolated or diverse from the greater masses of data are known as outliers. Historically, these outliers are handled as not representative, noisy or bogus data. However, in model building, it is know beforehand that all the available data represent the currently best part of the population. Therefore, no points must be disregarded. Instead, these outliers are essential, as they represent unexplored or recently discovered areas of the current Pareto-optimal front. That is why they should not only be preserved but, perhaps, even reinforced. A model-building algorithm that primes outliers might actually accelerate the search process and alleviate the rate of the exponential dimension-population size dependency.

It can be argued that the root cause that makes most standard methods to disregard outliers can be traced to the error-based learning that take place in those methods. In that

type of learning a dataset-wise error is minimized and because of that infrequent or poorly represented elements are sacrificed in order to achieve a better overall error.

The loss of diversity can be attributed to the above outliers' issue of model-building algorithms, among other possible causes. The repetitive application of an algorithm that disregards outliers tends to generate more individuals in areas of the search space that are more densely represented. Although there have been some proposals to circumvent this problem, we take the view that the ultimate solution is the use of an adequate algorithm.

Finally, for the model-building problem, there is no need for having the least complex accurate model of the data. However, most of the current approaches dedicate a sizable effort in finding the optimal model complexity by using minimum description length, structural risk minimization, Bayesian information criterion or other similar heuristics.

In high dimensional MOPs this model optimization will probably consume an excess of resources that would be more valuable if used by the rest of the algorithm. Instead a sufficient model can be inferred from the dimension of the search space and the data available. After all, in model building the main priority is not to obtain an optimal model (in terms of model dimension and structure) but a usable one. For instance, for cluster-based models its not required to find optimal amount of clusters just to find a "fair" amount such that the data set to be modeled is correctly covered.

1.5 Convergence Analysis and Stopping Criteria

When reviewing the current state of multi- and many-objective optimization it becomes noticeable the lack of a proper understanding of the nature of the evolutionary processes, in particular regarding convergence and *stopping criteria*.

Most soft-computing, heuristic, non-deterministic or numerical methods all have in common that they need a stopping criterion. The stopping criterion, which is usually a heuristic itself, is responsible for minimizing the wastage of computational resources by detecting scenarios where it makes no sense to continue executing the method.

The success or failure of any practical application relies heavily on not only the techniques applied but also the support methodologies, including the stopping criterion. Paradoxically, this is a matter that has often been overlooked by the community, probably because it plays a supporting part. This relegates the issue to an apparently secondary role. Consequently, the theoretical and practical implications concerning this topic have not yet been properly explored. Indeed, many real-world applications of theoretically outstanding methods may have underperformed due to an incorrect algorithm termination scheme.

The formulation of an effective criterion is particularly complex in the MOP case, as judging the optimization progress can turn out to be as complex as the optimization itself. In other types of problems, such as function approximation, pattern recognition or single-objective optimization, on the other hand, the axis can be used as a "zero" reference for progress measurement, as previously explained. This approach is unviable for MOPs since its solution is a set of points. Therefore, progress must be assessed in a relative manner using progress indicators rather than the actual solution set.

The most common practice to stop a MOEA in the multi–objective case is simply to halt the execution when the algorithm has reached a given number of iterations. A survey of papers that perform different sorts of comparative or experimental studies readily illustrates this situation.

Although this class of solution is probably viable for simple problems with two or three objective functions where the required number of iterations can be determined quite straightforwardly by trial and error, this procedure is computationally unaffordable for more complex problems.

This point is particularly applicable when dealing with MOPs with a relatively large number of objectives. In these problems, it is unfeasible to estimate the number of iterations required in order to converge to a given solution. Such estimation would require some knowledge of the nature of the problem or the assumption of some mathematical properties, which contradicts the central idea of evolutionary computation.

Probably on the above grounds, the formulation of an efficient stopping criterion for MOEAs and other MOP optimizers has been left aside, although it has been repeatedly named as one of the key topics in need of proper attention in the research area (Coello Coello, 2000, 2004).

1.6 Thesis Objectives

This thesis is mainly concerned with the two open issues described above; that is,

- the comprehension and solution of the model-building issues of current MOEDAs, and;
- the formulation of stopping criteria for multi-objective optimizers.

With regard to the first issue, we argue about what elements of MOEDAs should be modified in order to achieve a substantial improvement on their performance and scalability. However, in order to supply a solid ground for that discussion, some other elements are to be discussed as well. In particular, we:

- sketch the supporting theoretical corpus and the fundamentals of MOEA and MOEDA algorithms;
- analyze the scalability issue of MOEAs from both theoretical and experimental points of view;
- discuss the possible directions of improvement for MOEAs' scalability, presenting the current trends of research;
- give reasons of why EDAs can be used as a foundation for achieving a sizable improvement with regard to the scalability issue;
- examine the model-building issue in depth, hypothesizing on how it affects MOEDAs performance, and;

- propose a novel model-building algorithm, the *model-building growing neural gas* (MB-GNG); which fulfill the requirements for a new approach derived from the previous debate, and;
- propose a novel MOEDA, the *multi-objective neural estimation of distribution algorithm* (MONEDA), that is constructed on top of MB-GNG.

Theoretical discussions and algorithm proposals are experimentally contrasted with current state-of-the-art approaches when required.

The formulation of a strategy for stopping multi-objective optimizers became obvious and necessary as this thesis was developed. The lack of an adequate stopping criterion rendered any experimentation that had to do with many objectives a rather cumbersome task. That is why it was compulsory to deal with this issue in order to proceed with further studies. In this regard we:

- provide an updated and exhaustive state-of-the-art of this matter;
- examine the properties and characteristics that a given stopping criterion should exhibit;
- put forward a new stopping criterion, denominated MGBM, after the authors last names, that has a small computational footprint, and;
- we experimentally validate MGBM in a set of experiments.

1.7 Structure of the Thesis

This document is structured in four conceptual blocks. First, Chapter 2 introduces the theoretical foundations of this work. It presents the theoretical building blocks and the state the art of the matters that have to do with the thesis. This chapter provides a formal introduction to multi-objective optimization and presents evolutionary algorithms with special emphasis on multi-objective optimization. It also discusses the nature of many-objective problems and the current approaches to it. As a consequence, MOEDAs are described. A brief state-of-the-art of MOEDAs is then presented with special attention to model building. Finally, the convergence and stopping criteria issue is presented in detail.

Having thoroughly presented the necessary materials, the proposals put forward as part of this thesis are then conveyed. In particular, Chapter 3 discusses model building in MOEDAs, why we think that current approaches are not satisfactory and how to address this problem. Subsequently, Chapter 4 presents a modified growing neural gas network suited for model building (MB-GNG) that is embedded in the multi-objective neural EDA (MONEDA). MONEDA is a novel algorithm devised as a proof of concept and start point for further developments in this field. Also in this block we deal with the stopping criteria issue. Chapter 5 discusses this matter, introducing a novel stopping criterion MGBM.

The proposals introduced in the previous chapters need to be verified from an experimental point of view. That is the purpose of the next set of chapters. First, MGBM stopping criterion should be validated first, as it is meant to be used in the remaining experiments.

Chapter 6 contains a set of experiments carried out with this purpose. Chapter 7 compares MB-GNG with similar approaches. The results of this chapter are complemented with those exposed in the subsequent one, Chapter 8, in which MONEDA is compared with state of the art algorithms.

Finally, Chapter 9 summarizes the results obtained in the thesis and outlines the future trends of work.

A number of complementary appendixes are included mainly for reference purposes. In particular,

- Appendix A lists the publications related to this work.
- Appendix B describes the statistical validation framework used in experiments of Chapters 7 and 8.
- Appendix C describe the multi-objective test problems used in the experiment chapters.
- Appendix D contains a set of plots that show the progression of results in time of the experiments of Chapter 8.

For further details regarding this thesis and its related publications please visit the author's web page (http://www.giaa.inf.uc3m.es/miembros/lmarti).

2

Fundamentals

> We will now discuss in a little more detail the Struggle for Existence.
>
> *Charles Darwin — On the Origin of Species, Ch.3.*

In this chapter we introduce the matters related to the multi-objective optimization problem (MOP). First, we provide a formal description of the MOP and the definitions of optimality of solutions. After that, we examine how this problem has been dealt in the mathematical context, outlining the classical approaches. Subsequently, we discuss how the quality of solutions is evaluated and the outcomes of different algorithms are compared.

Relying on those foundations we introduce single- and multi-objective evolutionary algorithms, their theoretical elements and main characteristics. We present a brief survey of the most relevant approaches. After that, we dive into how multi-objective EAs perform when confronted to many-objective problems. We analyze the main issues with these problems and hypothesize on possible solutions or alternatives.

As a result of this analysis, we examine the viability of applying multi-objective estimation of distribution algorithms for dealing with these problems. Therefore, we survey current MOEDAs and study the issues that might be hampering the application these algorithms. These arguments are used to introduce one of the proposals put forward in subsequent chapters.

Finally, we present the current status of the understanding of convergence and stopping criteria, with emphasis on the conditions that prompted the development of the stopping criterion described in Chapter 5.

2.1 Multi-Objective Optimization

A multi-objective optimization problem (MOP), can be defined, without loss of generality[1], as:

[1] A maximization problem can be posed as as $\min\ -\boldsymbol{F}(\boldsymbol{x})$.

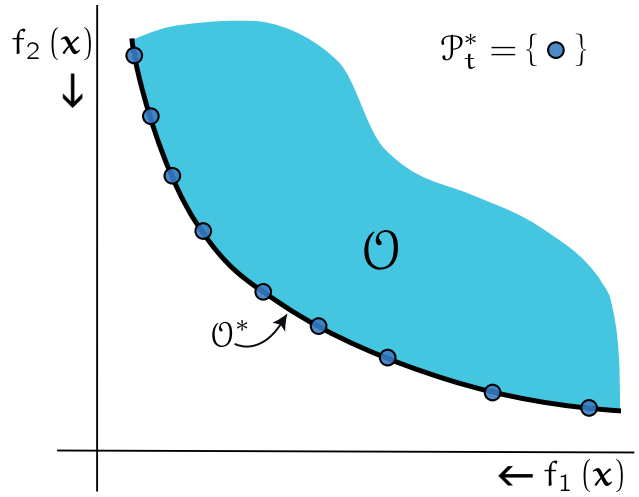


Figure 2.1: Graphical representation of a bidimensional multi-objective optimization problem. $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ are the functions to be minimized. The objective space $\mathcal{O}$ represent all the possible values of the functions. The Pareto-optimal front, $\mathcal{O}^*$, is represented as a curve. The solutions yielded by an optimizer, $\mathcal{P}_t^*$, are represented as blue dots.

Definition 2.1 (multi-objective optimization problem)

$$\left.\begin{array}{ll} \text{minimize} & \boldsymbol{F}(\boldsymbol{x}) = \langle f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x}) \rangle \,, \\ \text{subject to} & c_1(\boldsymbol{x}), \ldots, c_C(\boldsymbol{x}) \leq 0 \,, \\ & d_1(\boldsymbol{x}), \ldots, d_D(\boldsymbol{x}) = 0 \,, \\ & \text{with } \boldsymbol{x} \in \mathcal{D} \,, \end{array}\right\} \tag{2.1}$$

where $\mathcal{D}$ is known as the *decision set* or *search set*. The functions $f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})$ are the *objective functions*. If $M = 1$ the problem reduces to a single-objective optimization problem. The image set, $\mathcal{O}$, product of the projection of $\mathcal{D}$ via $f_1(\boldsymbol{x}), \ldots, f_M(\boldsymbol{x})$ is called *objective set* ($\boldsymbol{F} : \mathcal{D} \to \mathcal{O}$). These concepts are illustrated on Figure 2.1.

The characteristics of $\mathcal{D}$ and $\mathcal{O}$ largely define the tools used for solving the problem. For example, if $\mathcal{D} \subseteq \mathbb{R}^n$ we are facing a continuous optimization problem. On the other hand, if $\mathcal{D}$ has a discrete definition it is said to be a combinatorial optimization problem.

Finally, $c_1(\boldsymbol{x}), \ldots, c_C(\boldsymbol{x}) \leq 0$ and $d_1(\boldsymbol{x}), \ldots, d_D(\boldsymbol{x}) = 0$ express the constraints imposed on the values of $\boldsymbol{x}$. The subset resulting from imposing those constrains to $\mathcal{D}$ is known as *feasible set*, $\mathcal{S}$, and its corresponding subset of objective subset is the *feasible objective set*.

As stated previously in the introduction, in general terms, there is not a unique optimal solution to this class of problems. Usually, the solution to this type of problem is a set of trade-off points that contain equally good solutions, as shown in Figure 2.1. The optimality of a set of solutions can be defined relying on the so-called *Pareto dominance relation*:

Definition 2.2 (Pareto dominance relation) *For the optimization problem specified in (2.1) and having $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$, $\boldsymbol{x}$ is said to dominate $\boldsymbol{y}$ (expressed as $\boldsymbol{x} \prec \boldsymbol{y}$) iff $\forall f_j$, $f_j(\boldsymbol{x}) \leq f_j(\boldsymbol{y})$ and $\exists f_i$ such that $f_i(\boldsymbol{x}) < f_i(\boldsymbol{y})$.*

There are two more forms of Pareto dominance that, as will be used throughout the text, we will now introduce.

Definition 2.3 (Weak Pareto dominance) *Having $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$, $\boldsymbol{x}$ is said to weakly dominate $\boldsymbol{y}$ ($\boldsymbol{x} \preccurlyeq \boldsymbol{y}$) iff $\forall f_j$, $f_j(\boldsymbol{x}) \leq f_j(\boldsymbol{y})$.*

Definition 2.4 (Strict Pareto dominance) *Having $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D}$, $\boldsymbol{x}$ is said to strictly dominate $\boldsymbol{y}$ ($\boldsymbol{x} \prec\!\!\prec \boldsymbol{y}$) iff $\forall f_j$, $f_j(\boldsymbol{x}) < f_j(\boldsymbol{y})$.*

For a given set its *non-dominated subset* can be constructed using the Pareto dominance relation:

Definition 2.5 (non-dominated subset) *In problem (2.1) and having the set $\mathcal{A} \subseteq \mathcal{D}$. $\mathcal{A}^*$, the non-dominated subset of $\mathcal{A}$, is defined as*

$$\mathcal{A}^* = \{\boldsymbol{x} \in \mathcal{A} \,|\, \nexists \boldsymbol{y} \in \mathcal{A} : \boldsymbol{y} \prec \boldsymbol{x}\}\,.$$

Definition 2.6 (Pareto-optimal set) *The solution of problem (2.1) is the subset, $\mathcal{S}^*$, of non-dominated elements of $\mathcal{S}$.*

The subset $\mathcal{S}^*$ is known as the *Pareto-optimal set* or *efficient set*. Its image in objective set is called the *Pareto-optimal front*, $\mathcal{O}^*$,

Definition 2.7 (Pareto-optimal front) *For a Pareto-optimal set, $\mathcal{S}^*$, its corresponding Pareto-optimal front, $\mathcal{O}^*$ is defined as:*

$$\mathcal{O}^* = \{\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{x}); \forall \boldsymbol{x} \in \mathcal{S}^*\}\,. \tag{2.2}$$

This optimality condition can be further refined in order to encompass the binding between objectives. According to Definition 2.5, an element of $\mathcal{S}^*$ does not allow improvement of one objective function while retaining the same values on the others. Therefore, an improvement of in one objective can only be obtained at the expense of the deterioration of at least other one.

These trade-offs among objectives can be measured by computing the improvement in objective f_i, per unit decrease in a given objective f_j. In some situations such trade-offs can be unbounded, and, therefore, of no interest. The set of solutions with bounded objective function values is known as *proper Pareto-optimal set*. This set can be defined in different terms. For example,

Definition 2.8 (Geoffrion (1968) proper Pareto-optimal condition) *A feasible solution $\boldsymbol{x} \in \mathcal{S}$ is said to be properly Pareto-optimal if it is Pareto-optimal and $\exists b \in \mathbb{R}^+$, such that $\forall i = 1, \ldots, M$ and $\forall \boldsymbol{y} \in \mathcal{S}$ that satisfy $f_i(\boldsymbol{y}) < f_i(\boldsymbol{x})$, exists a f_j such that $f_j(\boldsymbol{x}) < f_j(\boldsymbol{y})$ bounded by*

$$\frac{f_i(\boldsymbol{x}) - f_i(\boldsymbol{y})}{f_j(\boldsymbol{y}) - f_j(\boldsymbol{x})} \leq b\,. \tag{2.3}$$

This not the only form of defining the proper optimality of solutions. In real-life applications, we will often encounter problems, where the feasible set is given implicitly by constraints,

$$\mathcal{S} = \{\boldsymbol{x} \in \mathbb{R}^n; (c_1(\boldsymbol{x}), \ldots, c_C(\boldsymbol{x})) \leq \mathbf{0}\} \ . \tag{2.4}$$

For such problems another definition of proper efficiency can be given, if the continuous differentiability of the objective functions $f_1, \ldots, f_M$ and constraint functions $c_1, \ldots, c_C$ can be asserted. This was first put forward first by Karush (1939) and later by Kuhn and Tucker (1951),

Definition 2.9 (Karush–Kuhn–Tucker proper optimality condition) *A solution $\boldsymbol{x} \in \mathcal{S}$ is said to be properly Pareto-optimal if it holds the Pareto-optimality condition of Definition 2.5 and $\nexists \boldsymbol{b} \in \mathbb{R}^n$ such that*

$$\forall i = 1, \ldots, M: \quad \nabla f_i(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{b} \leq \mathbf{0}; \tag{2.5}$$

$$\exists j = 1, \ldots, M \text{ such that}: \quad \nabla f_j(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{b} < \mathbf{0}; \tag{2.6}$$

$$\forall c_c(\boldsymbol{x}) = 0: \quad \nabla c_c(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{b} \leq \mathbf{0}. \tag{2.7}$$

These conditions are necessary when determining a Pareto-optimal solution. Furthermore, when all objective functions f_i are concave and $\mathcal{D}$ is a convex set this condition is sufficient as well. It has also been shown that, under some constraints, Definitions 2.8 and 2.9 are equivalent (Ehrgott, 2005).

If problem (2.1) has certain characteristics, e. g., linearity or convexity of the objective functions or convexity of $\mathcal{S}$, the efficient set can be determined by mathematical programming approaches (Branke et al., 2008). However, in the general case, finding the solution of (2.1) is an *NP*–complete problem (Bäck, 1996). In this case, heuristic or metaheuristic methods can be applied in order to have solutions of practical value at an admissible computational cost.1

Generally, an heuristic algorithm solving (2.1) yields a discrete local Pareto-optimal set, $\mathcal{P}^*$, that attempts to represent $\mathcal{S}^*$ as best as possible, although usually optimality can not be guarantied. The image of $\mathcal{P}^*$ in objective space, $\mathcal{PF}^*$, is known as the local Pareto-optimal front.

Even if $\mathcal{P}^*$ models $\mathcal{S}^*$ as best as possible, or even if $\mathcal{S}^*$ has been exactly found, the optimization process is not over. Elements of $\mathcal{S}^*$ are equally acceptable from a mathematical point of view, however, one solution, or, perhaps, a reduced set of solutions, should be selected in order to be realized in practice. The selection of the final solution(s) is carried out by a *decision maker* (DM). The DM is a person, group of persons, or a sort of automatic reasoning device that applies higher-order criteria in order to determine which of the elements of $\mathcal{P}^*$ are the chosen ones. Although this selection process is commonly carried out at the end of the optimization it can also take place during it as part of what has been called *interactive optimization methods* (Korhonen, 2005).

2.2 Evaluating Performance

Before diving into further analysis on the methods for carrying out a multi-objective optimization it is necessary to deal with one important issue: how to evaluate the quality of the

solutions yielded by such methods.

As new and better algorithms have been developed and more complex MOPs considered, the issue of performance assessment has become more and more relevant, up to the point that it has sprung into an independent research topic. The notion of algorithm performance involves both the quality of the solution found and the amount of resources required to generate that solution. Furthermore, the main issue with stochastic optimizers is that the relationship between quality and resource demands is not fixed, but, instead, may be described probabilistic terms. Therefore, every statement about the performance of these search algorithms is probabilistic in nature and pertaining only to its particular experimental context.

In the case of multi-objective optimization there is a particular complication. It involves at least two objectives being optimized and, consequently, the outcome of the optimization process is usually not a single solution but a set of trade-offs. This not only raises the question of how to define quality in this context, but also how to represent the outcomes of multiple runs in terms of a probability density function.

Early works that carried out comparative experiments limited themselves to visually compare the characteristics of plots of different non-dominated sets and fronts (Coello Coello, 2000). This solution has many evident shortcomings. The first is derived from the two or three-dimensional limit imposed by the plot representation. Similarly, as the properties of the Pareto-optimal sets or front became more and more complex the interpretation and comparison of results could become an "art appreciation" skill. Finally, it is impossible in practice to come up with any statistically valid statement on the nature of the results.

Therefore, two issues must be tackled in order to capture a proper picture of the quality of solutions: the formulation of quality indicators capable of expressing the adequacy of a given set of solutions, and; the selection of the correct statistical tools in order to combine different independent runs of the algorithm of the problem. Appendix B describes the strategy followed in the experiments carried out in this thesis.

Determining how good a given set of solutions, $\mathcal{P}^*$, is with regard to the Pareto-optimal set, $\mathcal{S}^*$, and, particularly a $\mathcal{PF}^*$ with regard to $\mathcal{O}^*$, is not only a key but also a particularly complex task. It necessarily implies a reduction from an M-dimensional space to a scalar value. Therefore, as in any dimensionality reduction, valuable information may be lost, leading to invalid conclusions. This point has been well documented by Zitzler et al. (2002b, 2003).

The outperformance of a set of solutions over another can be formalized on top of the Pareto dominance relation with an extension to sets:

Definition 2.10 *Having the sets $\mathcal{A}, \mathcal{B} \subseteq \mathcal{D}$ of candidate solutions to problem (2.1); $\mathcal{A}$ is said to dominate $\mathcal{B}$ ($\mathcal{A} \prec \mathcal{B}$) if $\forall \boldsymbol{x} \in \mathcal{A}\ \nexists \boldsymbol{y} \in \mathcal{B}$ such that $\boldsymbol{x} \prec \boldsymbol{y}$.*

As a consequence we can state when a set is better than other:

Definition 2.11 *Having $\mathcal{A}, \mathcal{B} \subseteq \mathcal{D}$. $\mathcal{A}$ is better than $\mathcal{B}$ ($\mathcal{A} \triangleleft \mathcal{B}$) if $\mathcal{A} \prec \mathcal{B}$ and $\mathcal{B} \not\prec \mathcal{A}$.*

This form of set comparison leave little room for analysis or interpretation, as a very limited amount of information can be extracted. For example if the sets overlap each other, it

is impossible to determine at what degree one overlaps the other. It is therefore desirable to quantify the difference in quality on a continuous scale. For instance, we may be interested in knowing how much better a set is with regard to other. This is crucial for the search process itself, and almost all algorithms for approximating the Pareto set make use of additional preference information, e. g., in terms of diversity measures.

Quality indicators have been devised for that purpose:

Definition 2.12 (quality indicator) *A quality indicator is a function $I : \mathcal{S} \to \mathbb{R}$ that assigns a real number to a given set of solutions $\mathcal{S} \subseteq \mathcal{D}$.*

In principle, one may consider any function I as an indicator. However, it is obvious that there are certain properties that need to be fulfilled in order to make the indicator useful.

There are five main criteria that are particularly important and should be kept in mind when applying an indicator:

- *Quantified aspect*: Different indicators are devised to expose different properties of the set being analyzed. These indicators can be grouped into three broad categories:
  1. distance from the elements of $\mathcal{PF}^*$ to their corresponding closest element of $\mathcal{O}^*$, which measures how close the solution is to the optima;
  2. distance from every element of $\mathcal{O}^*$ to its closest element of $\mathcal{PF}^*$, which complements the first class of indicators and expresses how well $\mathcal{PF}^*$ covers $\mathcal{O}^*$, and
  3. distribution of the elements of $\mathcal{P}^*$ and $\mathcal{PF}^*$, which gauges how well spread the elements of these sets are.
- *Monotonicity*: An indicator I is said to be Pareto-monotonic iff for any two sets of solutions if the one set dominates the other it implies that it is also has better indicator values, that is,
$$\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{D} : \mathcal{A} \preccurlyeq \mathcal{B} \Rightarrow I(\mathcal{A}) \leq I(\mathcal{B})\,; \tag{2.8}$$
assuming that smaller values of the indicator are better. This property guarantees that I is consistent with the partial order of derived from the weak Pareto dominance relation. However, a set that has the same indicator value as the Pareto-optimal set not necessarily contains only Pareto-optimal solutions. A stronger condition is needed to ensure this: the *strict Pareto monotonicity*,
$$\forall \mathcal{A}, \mathcal{B} \subseteq \mathcal{D} : \mathcal{A} \prec \mathcal{B} \Rightarrow I(\mathcal{A}) < I(\mathcal{B})\,. \tag{2.9}$$
- *Scaling invariance*: Objective functions frequently take values in different ranges. In this context, it may be desirable that an indicator is not affected by this difference in ranges or, alternatively, is resilient to any type of scaling. Scaling invariant indicators usually extract their information from the dominance relation among solutions, and not from their plain objective function values.
- *Computational requirements*: An important thing to take into account when choosing a quality indicator is its computational complexity and cost. These properties mainly depends on the dimensions of the decision and objective spaces and the number of elements in the set being processed. This issue should be taken into account particularly

when using the indicator as part of the optimization process and, therefore, it is being intensively computed.

- *Parameterization*: Many indicators require some sort of parameter to be set beforehand. The correct selection of these parameters has a direct impact on the accuracy and reproducibility of results. In practical cases the determination of these parameters is non-trivial as the properties of the problem are unknown. Furthermore, some indicators require a known Pareto-optimal set in order to carry out their computation. Obviously, this set is not available in many practical situations.

There are a rather large number of indicators. We will now present the ones that are of interest as they will be take part in subsequent discussions and in the experiments carried out. We refer the interested reader to (Zitzler et al., 2008) and (Coello Coello et al., 2007, ch.5) for a survey on this matter. Quality indicators were conceived to evaluate the performance of a given optimizer. However, they can be reformulated to form part of a stopping criterion framework. Relatively recently their use also been extended to the optimization process itself, as their are used for determining search direction.

2.2.1 Hypervolume Indicator

The *hypervolume indicator*, $I_{\text{hyp}}(\mathcal{A})$, (Knowles et al., 2006a; Knowles, 2002; Zitzler et al., 2007, 1999) computes the volume of the region, H, delimited by a given set of points, $\mathcal{A}$, and a set of reference points, $\mathcal{N}$.

$$I_{\text{hyp}}(\mathcal{A}) = \text{volume}\left(\bigcup_{\forall \boldsymbol{a}\in\mathcal{A};\forall \boldsymbol{n}\in\mathcal{N}} \text{hypercube}(\text{a},\text{n})\right). \tag{2.10}$$

Therefore, larger values of the indicator will correspond to better solutions.

The hypervolume indicator is also known as the $\mathcal{S}$ metric or the Lebesgue measure. It has many attractive features that had favored its application and popularity. In particular, it is the only indicator that has the properties of a metric and the only to be strictly Pareto monotonic (Fleischer, 2003; Zitzler et al., 2003). Because of these properties this indicator has been used not only for performance assessment but also as part of some evolutionary algorithms (see Section 2.7 for details).

To measure the absolute performance of an algorithm the reference points should ideally be *nadir points*. These points are the worst elements of $\mathcal{O}$, or, in other words, the elements of $\mathcal{O}$ that do not dominate any other element. To contrast the relative performance of two sets of solutions, though, one can be used as the reference set. These matters are further detailed in (Knowles et al., 2006a; Zitzler et al., 2002b).

Having $\mathcal{N}$, the computation of the indicator is a non-trivial problem. Indeed, its determination is known to be computationally intensive, thus rendering it unsuitable for problems with many objectives.

A lot of research has focused on improving the computational complexity of this indicator (Beume, 2009; Beume and Rudolph, 2006; Fonseca et al., 2006; While et al., 2005, 2006). The exact computation of the algorithm has been shown to be #*P*-hard (Bringmann

and Friedrich, 2010) in the number of objectives. #*P* problems are the analogous of *NP* for counting problems (Papadimitriou, 1994). Therefore, all algorithms calculating a hypervolume must have an exponential runtime with regard to the number of objectives if *P*≠*NP*, something that seems to be true (Deolalikar, 2010).

According to the most recent results, the indicator is currently known to be $O(n \log n + n^{M/2})$ (Beume, 2009) for more than three objectives ($M > 3$); $O(n \log n)$ for $M = 2,3$ (Fonseca et al., 2006).

One alternative is to circumvent the complexity hurdle is to apply estimation algorithms capable of yielding an approximation of the indicator at a more convenient temporal cost. Monte Carlo sampling (Bader and Zitzler, 2008; Bringmann and Friedrich, 2009, 2010) and *k*-greedy strategy (Zitzler et al., 2010) have been applied with success.

This indicator can also be used to measure the progress of an algorithm as the evolution proceeds. In order to do this the indicator should be transformed into a relative formulation, as proposed by the binary hypervolume indicator (Knowles et al., 2006b):

$$I_{\text{hyp}}(\mathcal{A}, \mathcal{B}) = I_{\text{hyp}}(\mathcal{A}) - I_{\text{hyp}}(\mathcal{B}) \ . \tag{2.11}$$

Substituting $\mathcal{A}$ and $\mathcal{B}$ by the non-dominated elements of the current and the previous iteration, $\mathcal{PF}_t^*$ and $\mathcal{PF}_{t-1}^*$, respectively, the indicator can be expressed as

$$I_{\text{hyp}}(t) = I_{\text{hyp}}(\mathcal{PF}_t^*) - I_{\text{hyp}}(\mathcal{PF}_{t-1}^*) \ . \tag{2.12}$$

2.2.2 Epsilon Indicators

Epsilon indicators (Knowles et al., 2006b; Zitzler et al., 2003) are a set of performance indicators that rely on the epsilon dominance concept. These indicators were proposed to measure how close the current non-dominated solution individuals front, $\mathcal{PF}_t^*$, is to the Pareto-optimal front, $\mathcal{O}^*$.

Epsilon dominance is a relaxed version of the Pareto dominance relation. This ϵ-dominance relation is presented in Definitions 2.13 and 2.14 in additive and multiplicative terms, respectively.

Definition 2.13 (Additive ϵ-dominance relation) *For the optimization problem specified in (2.1) and having $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{D}$, $\boldsymbol{x}_1$ is said to additively ϵ-dominate $\boldsymbol{x}_2$ (expressed as $\boldsymbol{x}_1 \preccurlyeq_{\epsilon+} \boldsymbol{x}_2$) iff $f_j(\boldsymbol{x}_1) \leq \epsilon + f_j(\boldsymbol{x}_2)$.*

Definition 2.14 (Multiplicative ϵ-dominance relation) *For problem (2.1) and having $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{D}$, $\boldsymbol{x}_1$ is said to multiplicatively ϵ-dominate $\boldsymbol{x}_2$ ($\boldsymbol{x}_1 \preccurlyeq_{\epsilon\cdot} \boldsymbol{x}_2$) iff $f_j(\boldsymbol{x}_1) \leq \epsilon f_j(\boldsymbol{x}_2)$.*

The additive epsilon indicator, $I_{\epsilon+}$, is a relative indicator that expresses the minimum value of ϵ that is necessary to make a set $\mathcal{A}$ ϵ-dominate a set $\mathcal{B}$, that is,

$$I_{\epsilon+}(\mathcal{A}, \mathcal{B}) = \inf_{\epsilon \in \mathbb{R}} \{\forall \boldsymbol{y} \in \mathcal{B},\ \exists \boldsymbol{x} \in \mathcal{A} \text{ such that } \boldsymbol{x} \preccurlyeq_{\epsilon+} \boldsymbol{y}\} \ . \tag{2.13}$$

The multiplicative version is defined in similar terms,

$$I_{\epsilon\cdot}(\mathcal{A},\mathcal{B}) = \inf_{\epsilon\in\mathbb{R}} \{\forall \boldsymbol{y}\in\mathcal{B},\ \exists \boldsymbol{x}\in\mathcal{A} \text{ such that } \boldsymbol{x} \preccurlyeq_{\epsilon\cdot} \boldsymbol{y}\}\ . \tag{2.14}$$

The value of the indicators is to be minimized. From the two, the most commonly used is the additive version.

If $I_{\epsilon+} < 0$, then $\mathcal{A}$ dominates $\mathcal{B}$. It is Pareto-monotonic but not strictly. On the other hand, it can be computed in time $O(M|\mathcal{A}||\mathcal{B}|)$, a low complexity, when compared with the hypervolume indicator. Nevertheless, it has the weakness that, for some cases,

$$|I_{\epsilon+}(\mathcal{A},\mathcal{B})| \neq |I_{\epsilon+}(\mathcal{B},\mathcal{A})|\ . \tag{2.15}$$

A reformulation aimed to overcome this asymmetry would imply an increment in the order of the algorithm complexity.

$I_{\epsilon+}$ can be directly applied to progress assessment by substituting $\mathcal{A}$ and $\mathcal{B}$ by the Pareto-optimal sets of two consecutive iterations, $\mathcal{PF}^*_t$ and $\mathcal{PF}^*_{t-1}$.

2.2.3 Pareto-optimal Front Coverage Indicator

The *Pareto-optimal front coverage indicator*, $I_{\text{cov}}(\mathcal{A})$, (Bosman, 2003) complements the first two indicators as it describes how close the elements of $\mathcal{O}^*$ are to their closest counterpart in the current $\mathcal{PF}^*_t$. In this case, smaller values of the indicator are desired.

I_{cov} is a combined measure of how close the solutions are to the Pareto-optimal front are while at the same time assessing how diverse these solutions are and how well they are distributed along $\mathcal{O}^*$.

If the Pareto optimal front is continuous, a correct formulation of this indicator calls for a line integration over $\mathcal{O}^*$. A simpler approach assumes that $\mathcal{O}^*$ is discrete. In this case the indicator is formulated as

$$I_{\text{cov}}(\mathcal{O}^*,\mathcal{P}^*_t) = \frac{1}{|\mathcal{O}^*|}\sum_{\boldsymbol{x}\in\mathcal{O}^*} \min_{\boldsymbol{y}\in\mathcal{P}^*_t} d(\boldsymbol{x},\boldsymbol{y})\,. \tag{2.16}$$

Where $d(\cdot)$ is the Euclidean distance

$$d(\boldsymbol{x},\boldsymbol{y}) = \sqrt{\sum_{j=1\ldots M}(x_j - y_j)^2}. \tag{2.17}$$

2.3 Evolutionary Algorithms

In rough terms, an evolutionary algorithm can be characterized by how it implements a set of processes (see Figure 2.2 for a diagram), in particular,

- *Mating selection*: that establishes a partial order of individuals in the population using their fitness function value as reference and determines the degree at which individuals in the population will take part in the generation of new (offspring) individuals.

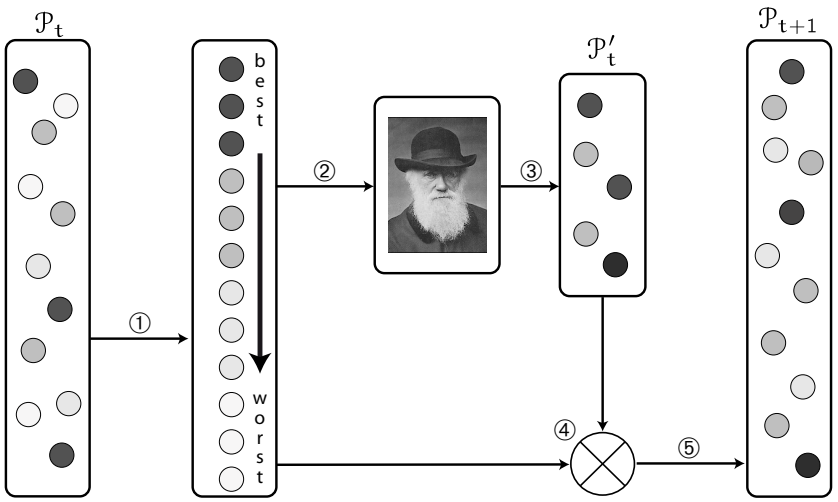


Figure 2.2: Simplified schematic representation of an evolutionary algorithm iteration. Individuals of the current population, $\mathcal{P}_t$, have their fitness values calculated (1). Evolutionary operators are applied to the population by taking into account the fitness of each individual (2). This causes that an offspring population, $\mathcal{P}'_t$, containing new individual to be generated (3). The original and offspring populations are merged (4), producing the next iteration population, $\mathcal{P}_{t+1}$ (5).

- *Offspring generation (variation)*: which applies a range of evolutionary operators to synthesize offspring individuals from the current (parent) population. This process is supposed to prime the fittest individuals so they play a bigger role in the generation of the offspring.

- *Parents and offspring combination (environmental selection)*: that merges the parent and offspring individuals to produce the population that will be used in the next iteration. This process often involves the deletion of some individuals using a given criterion in order to keep the amount of individuals bellow a certain threshold.

In single-objective optimization the determination of the fitness of an individual is straightforward, as the value of the function being optimized can be directly used as fitness.

The selection process should bias better ranked individuals with regard to less performing ones to prompt the first ones to take a more active role in the synthesis of offspring. There are different strategies to carry this out. For example, the *fitness-proportionate selection* assigns a higher selection probability to individuals with higher fitness. It is like a casino roulette in individuals are assigned circular sectors proportionate to their fitness. Similarly, *tournament selection* repeatedly selects the best individual of a randomly chosen population subset. *Truncation selection* directly extracts a population subset that contains the best individuals. This subset is generally the top third or half of the population. There are, of course, many other selection strategies. The choice of selection method depends on the nature of the problem and it is tightly connected with the form of population ranking.

The generation of new individuals give EAs the ability of exploring the search space. Some of the forms of generating new individuals have a natural inspiration. In this class we find the *crossover operator* that interchanges parts of the chromosomes of two individuals.

The nature of this operation depends of the properties of the chromosomes. For example, it is rather simple to interchange parts of binary chromosomes, but it is not trivial to define the same operation for real-valued ones.

Mutation is another common form of creating new individuals that have a counterpart in the natural world. In this case a given element of a chromosome is selected at random and modified in a certain way. For example, in binary chromosomes this implies complementing the bit stored at the selected element. If more complex representations are used, the nature of the mutation operator becomes harder to define.

It is often argued that the crossover operator *exploits* the knowledge of the fitness landscape in a form comparable to a local search. On the other hand, mutation *explores* the search set by proving unvisited areas.

There are other methods for creating new individuals. For example, in differential evolution (Storn and Price, 1997) three individuals are used to modify the chromosome of a fourth one; while in estimation of distribution algorithms, which we will discuss in-depth in subsequent parts of this document, a statistical model of the selected population is used.

After having the new individuals its necessary to have a certain policy for determining what individuals of the parent population should be discarded, assuming that it is desired to keep a constant population size. An approach similar to *truncation selection* removes the worst individuals an replaces them with the new ones. An alternative is to randomly select the individuals to be removed. In order to not to loose valuable individuals (i. e. the individuals with better fitness) some algorithms ensure that the best subset of the population is preserved across iterations. This is called *elite preservation*, and is present in most modern evolutionary approaches.

Besides these processes, an EA is also characterized by the way each individual encodes the point of the search space it represents and the selection of evolutionary operators used. Sometimes the decision regarding these last two features depends on the particular problem being solved. However, each EA paradigms might also establish some restrictions or limits regarding the aforementioned processes.

EAs, as many other heuristic, non–deterministic or numerical methods, require a stopping criterion that decides when the execution should be ended. This matter is further analyzed on the forthcoming Section 2.11.

There have been diverse attempts to provide a unifying formalization for evolutionary computation (see, for example, De Jong (2006)). However, these approaches have not prospered mostly because of the diverse nature of the approaches. A clear dissection of the EC field into and ordered taxonomy can be a frustrating task because of the interrelation and hybridization of methods and the fuzzy boundaries between approaches. Still, some main classes can be distinguished, in particular, evolutionary programming (Fogel et al., 1966), evolutionary strategies (Rechenberg, 1973; Schwefel, 1977), genetic algorithms (Goldberg, 1989) and genetic programming (Koza, 1992).

Evolutionary approaches have seen successful applications in multiple areas, ranging, for example, form economy and finance (Cubo et al., 2005; Mochón et al., 2008), cryptography (Isasi and Hernandez, 2004), radio network design (Mendes et al., 2009), bioinformatics (Santana et al., 2008), robotics (Santana and Correia, 2011), among many others.

2.4 Multi-Objective Evolutionary Algorithms

Single objective EAs have been extrapolated with success to the multi-objective scenario. This has led to what has been called *multi-objective optimization evolutionary algorithms* (MOEAs). Work on this subject started with the vector evaluated genetic algorithm (VEGA) (Schaffer, 1985) and the reflections found in (Goldberg, 1989).

In classical (single objective) EAs there is only one objective function to be minimized or maximized. Thanks to that, it is straightforward to use that function for determining the fitness of individuals. Finding this scalar indicator, as already commented in Section 2.2, is a complex matter since, as in any dimensionality reduction, relevant information may be lost. Furthermore, a MOP solver is not only expected to yield solutions as close as possible to the Pareto-optimal front. Its solutions should also be as diverse as possible, therefore offering a good coverage of $\mathcal{O}^*$.

Because of these reasons the ranking of individuals is one of the key issues in the MOEAs' field of research. The strategies that have been proposed to circumvent this problem can be grouped in three classes:

- *Objective function aggregation*: where objective values are combined using a weighted aggregation function of either linear or non-linear nature.
- *Pareto-based ranking*: that generate an ordering of the population individuals relying on the domination relation.
- *Indicator-based ranking*: which use the performance indicators like the ones introduced in Section 2.2 and originally meant for assessing MOP optimizer's performance.

The Pareto-based ranking is, so far, the most popular approach. The objective function aggregation is not very used as it yields a unique solution, instead of a set of them. Similarly, if the aggregation is of linear nature it has problems when faced with a concave Pareto-optimal front. The indicator-based ranking is a relatively new approach and still need further research. Although it is a promising direction of work it these strategies seem to be very intensive in terms of computational resources.

2.5 Objective Function Aggregation

A common way of synthesizing that aggregate function is via a weighted sum, leading to the an scalarized version of (2.1),

$$\left.\begin{array}{ll} \text{minimize} & F(\boldsymbol{x}) = \sum_{m=1}^{M} w_m f_m(\boldsymbol{x})\,, \\ \text{subject to} & c_1(\boldsymbol{x}), \ldots, c_C(\boldsymbol{x}) \leq 0\,, \\ & d_1(\boldsymbol{x}), \ldots, d_D(\boldsymbol{x}) = 0\,, \\ & \text{with } \boldsymbol{x} \in \mathcal{D}\,. \end{array}\right\} \tag{2.18}$$

Usually weights w_m are chosen in such way that $\sum_m w_m = 1$, for practical reasons. These weights can be set by the DM in order to express a deliberate interest in certain objectives.

Another approach is to concentrate on only one of M objectives while all the other objectives are transformed into constraints by introducing upper bounds. This scalarization is called epsilon constraint method and is given by

$$\left.\begin{array}{ll} \text{minimize} & F(\boldsymbol{x}) = f_k(\boldsymbol{x})\,, \\ \text{subject to} & f_m(\boldsymbol{x}) \leq \epsilon_m, m = 1 \ldots M, m \neq k\,, \\ & c_1(\boldsymbol{x}), \ldots, c_C(\boldsymbol{x}) \leq 0\,, \\ & d_1(\boldsymbol{x}), \ldots, d_D(\boldsymbol{x}) = 0\,, \\ & \text{with } \boldsymbol{x} \in \mathcal{D}\,. \end{array}\right\} \tag{2.19}$$

The case of the weighted sum method has been particularly studied. It has been shown that it can only be safely applied to problems with a convex Pareto-optimal front. Yet problems arising in real-world applications are often non-convex. Furthermore, there are many cases when tests regarding the shape of the front can not be investigated. On the positive side, scalarization methods have a low computational footprint and therefore can handle complex problems.

2.6 Pareto-Based Approaches

The first MOEAs proposed were of non-elitists nature. In this group we can find the multi-objective genetic algorithm (MOGA) (Fonseca and Fleming, 1993a,b), the non-dominated sorting genetic algorithm (NSGA) (Srinivas and Deb, 1994) and the niched–Pareto genetic algorithm (NPGA) (Horn et al., 1994) among others. The last three algorithms apply non-dominated classification in the population, with variations in the type of selection operators.

Elitist approaches to MOEA are the most recent techniques. Among them, we can mention the improved NSGA (NSGA-II) (Deb et al., 2002), the strength Pareto evolutionary algorithm (SPEA) (Zitzler and Thiele, 1998), the improved SPEA (SPEA2) (Zitzler et al., 2002a), Pareto-archived evolution strategy (PAES) (Knowles and Corne, 1999, 2000) and the Pareto envelope-based selection algorithm (PESA) (Corne et al., 2001, 2000), among many others.

2.6.1 Pareto-Archived Evolution Strategy (PAES)

Knowles and Corne (1999, 2000) proposed a MOEA called *Pareto-archived evolution strategy* (PAES). In PAES every individual of the population generates, by mutation, its offspring. The offspring is compared with the parent. If the offspring dominates its parent, the offspring substitutes its parent. On the other hand, if the parent dominates the offspring, the offspring is discarded and another mutated solution is generated and the comparison process is repeated. If the offspring and the parent do not dominate each other, a comparison set of previously non-dominated individuals is used for the comparison.

In order to keep population diversity along the Pareto-optimal front, an archive of non-dominated solutions is used. A new generated offspring is compared with the archive to verify if it dominates any member of the archive. If yes, then the offspring enters the archive and is accepted as a new parent.

Dominated solutions are removed from the archive. If the offspring does not dominate any member of the archive, both parent and offspring are checked. The one situated in a less

crowded region of the archive is kept and if the archive is not full then both solutions can be kept and another solution from a most crowded region is removed.

2.6.2 Improved Strength Pareto Evolutionary Algorithm (SPEA2)

The original SPEA implements elitism by preserving an external population. This external population stores a fixed amount of non-dominated solutions discovered since the beginning of the simulation. After every iteration of the algorithm, if a new non-dominated solution is found it is compared with the ones present in the external population to preserve the best solutions.

SPEA goes beyond than just keeping an elite set of solutions. It uses the solutions stored along with the dominated solutions in all genetic operations with the hope of inducing a better performance of the search in the solution space.

Although SPEA has produced a number of relevant results it has been pointed out that it has some potential weaknesses. SPEA2 was proposed as an attempt to overcome the limitations of SPEA. It keeps the overall scheme of its predecessor but, in contrast to SPEA, SPEA2 uses a fine-grained fitness assignment strategy that incorporates density information. Furthermore, the external population has a fixed size; therefore, whenever the number of non-dominated solutions is less than the predefined archive size, the archive is filled up by dominated individuals. Finally, the clustering technique used to prune the external population has been replaced by an alternative truncation method that has similar features but does not miss boundary points.

2.6.3 Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II)

The NSGA-II algorithm is an improvement over NSGA. There are two key concepts in the NSGA family: fast non-dominated sorting of the population and a crowding distance calculation for maintaining diversity in the population.

The crowding distance considers the size of the largest cuboid enclosing each individual without including any other of the population. This feature is used to keep diversity in the population and points belonging to the same front and with higher crowding distance are assigned a better fitness than those with lower crowding distance, avoiding the use of the fitness sharing factor.

NSGA-II introduces a faster algorithm to sort the population that takes $O(mk^2)$ computations, instead of the original $O(Mk^3)$ of NSGA, where M is the number of objectives and k is the number of population members. NSGA-II also incorporates an elitism scheme for preserving candidates solutions.

2.6.4 Pareto Envelope-Based Selection Algorithm (PESA)

The *Pareto envelope-based selection algorithm* (PESA) (Corne et al., 2001, 2000) is a hybrid algorithm between PAES and SPEA. It uses a small internal population and a larger external population (archive), where non-dominated solutions found in the main population are stored using a hyper-grid based scheme.

Objective space is divided into several regions, and the selection mechanism is based on the degree of crowding in different regions of the archive. Empirical studies have demonstrated that PESA performs better than PAES in several test functions.

2.7 Indicator-Based Approaches

Ranking and comparing individuals via the dominance relation disregards the distance between individuals, as it is a crisp criterion. Therefore, individuals close to each other are treated similarly as those far apart, as there is not such thing as "amount of domination" information. This lack of granularity implies that larger populations are necessary in order to gather sufficient information to guide the search.

Performance indicators, as the ones described in Section 2.2 can be used to provide that missing information. The hypervolume indicator, in particular has been the most popular option so far, as it is monotonic with respect to the Pareto dominance relation. This had led to the proposals of a number of MOEAs that exploit this idea.

2.7.1 Indicator-Based Evolutionary Algorithm (IBEA)

Zitzler and Künzli (2004) proposed a general evolutionary framework in which different indicators can be integrated. The algorithm assigns the fitness of an individual as an aggregation of the relative hypervolume of it with regard to the rest of the population,

$$\text{fitness}(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in \mathcal{P} \setminus \{\boldsymbol{x}\}} -\exp\left(I(\boldsymbol{x}, \boldsymbol{y})/\kappa\right), \tag{2.20}$$

where κ is a fitness scaling factor.

A subset of the elements with the lowest fitness values are then removed from the population $\mathcal{P}$. A binary tournament selection with replacement takes place to create a temporary matting poll. Recombination and mutation is applied to the pool to generate the offspring population to be used in the next iteration.

IBEA has seen diverse improvements, as, for example, the self-adaptation of the scaling parameter κ, or the application of a non-dominated ranking and only computing (2.20) in cases of non-comparable individuals.

IBEA has allowed to discover an issue regarding these approaches: how to determine the subset of the population that has the lowest hypervolume contribution.

2.7.2 $\mathcal{S}$ Metric Selection Evolutionary Multiobjective Optimization Algorithm (SMS-EMOA)

SMS-EMOA (Beume et al., 2007) is an steady-state algorithm. That means that, in every iteration, only one individual is created, only one has to be deleted from the population in each generation. The hypervolume is no computed exactly. Instead, the k-greedy strategy is employed. These decisions were made in the hope of tackling the high computational demands of computing the hypervolume.

Perhaps the key element of SMS-EMOA is the method for determining which element of the population will be substituted by the offspring. This is done, by applying a non-domination ranking. From the elements that are dominated by the rest of the population, $\mathcal{P}_{\text{worst}}$, one is selected such that it has the minimum contribution to the hypervolume of the set

$$\boldsymbol{x}_{\text{rem}} = \arg\min_{\boldsymbol{x}\in\mathcal{P}_{\text{worst}}} I\left(\mathcal{P}_{\text{worst}}\right) - I\left(\mathcal{P}_{\text{worst}} \setminus \{\boldsymbol{x}\}\right). \tag{2.21}$$

$\boldsymbol{x}_{\text{rem}}$ is to be removed from the population and substituted by a new individual generated by the usual variation operators.

2.7.3 Hypervolume Estimation Algorithm for Multiobjective Optimization (HypE)

HypE (Bader and Zitzler, 2011) addresses the issue of the computational cost of the hypervolume indicator by proposing Monte Carlo sampling method that approximates the value of the hypervolume at a lower computational cost. This approximation, even when it might not be excessively accurate is usable for the fitness assignment purpose.

HypE also proposed a strategy for environmental selection that guaranties the deletion from the population of the worst elements in terms of hypervolume contribution.

2.8 Many-Objective Problems

One topic that remains not properly dealt with inside the MOEA scope is the scalability of the algorithms (Coello Coello, 2006a,b). For these algorithms their scalability issue is exposed regarding two quantities:

1. the dimension of the decision set, $\mathcal{D}$, that is, the number of variables that take part of the problem, and;

2. the dimension of the objective set, $\mathcal{O}$, or in other words, the number of objective functions to be optimized.

A critical quantity is the dimension of the objective space as it has been experimentally shown to have an exponential relation with the optimal size of the population (see Khare et al. (2003); Praditwong and Yao (2007); Purshouse and Fleming (2007) and Deb (2001, pp. 414–419)). This fact implies that, with the increase of the number of objective functions an optimization algorithm needs an exponential amount of resources made available to it.

As mentioned in the introduction of this document, there is a class of MOP that is particularly appealing because of their inherent complexity: the so-called *many-objective problems* (Purshouse and Fleming, 2007). These are problems with a relatively large number of objectives (normally, four or more).

2.8.1 Improving the Fitness Assignment

This question can be reduced to the problem of how to handle a relatively small population in which non-dominated and dominated individuals are not adequately balanced and represented. This fact causes approaches based on Pareto dominance to be rendered useless. Therefore, in order to achieve a sizable improvement in the scalability of these algorithms, it is essential to arm them with an efficient, scalable and robust fitness assignment function that promotes search and pushes the population towards newly found sub-optimal zones in objective space.

The issue of finding a better fitness assignment function is a complex one. As previously discussed, as dimensions grow, there is an exponential explosion in the amount of resources required. This growth implies that mutual comparison and/or sorting processes that take place as part of the fitness assignment become very time consuming. One possible solution is to bypass the exponential relation, but this will probably lead to a situation where there is a high degree of non-domination of individuals and therefore it is impossible to find a correct direction for the search.

There have been a number of works (Bader et al., 2010; Bader and Zitzler, 2008; Basseur and Zitzler, 2006a,b; Brockhoff and Zitzler, 2007b; Zitzler and Künzli, 2004) that propose the use of performance indicators, in particular the hypervolume indicator (Zitzler et al., 2007, 1999), as fitness assignment functions.

Wagner et al. (2007) reported good results by IBEAs for many-objective problems. Since IBEAs do not use Pareto dominance, their search ability is not severely deteriorated by the increase in the number of objectives. One difficulty in the application of IBEAs to many-objective problem is a large computation cost for hypervolume calculation.

This is a promising line of research as these approaches might overcome the situations where most of the population is non-dominated and a direction of search cannot be found. However, the computation of this indicator has been shown to be very computationally complex (see Sections 2.2 and 2.7 for further details). Still, some recent developments in this direction (Bader, 2010; Beume, 2009; Beume and Rudolph, 2006) have opened a door for their application in high-dimensional problems.

2.8.2 Reduction of Objective Functions

As it was mentioned before, there has been a number of works (Brockhoff et al., 2008; Brockhoff and Zitzler, 2007a; Deb and Saxena, 2005, 2006) directed towards the reduction of the number of objective functions to a minimum and therefore towards the mitigation of the complexity of a problem. Although these works provide a most useful tool for alleviating the burden of a given problem they do not ultimately address the essential issue: how to create MOEDAs capable of efficiently solve high–dimensional problems.

2.8.3 Incorporating Learning in the Optimization

Another viable approach is to employ cutting-edge evolutionary algorithms that would deal with high-dimensional problems more efficiently. The incorporation of learning as part of

the search processes has been nominated as a viable solution (Corne, 2008). One of the forms of including learning as part of the search process is to apply estimation of distribution algorithms.

One of the forms of carrying out this integration is the application of *estimation of distribution algorithms* (EDAs) (Baluja, 1994; Larrañaga and Lozano, 2002; Lozano et al., 2006; Mühlenbein and Paaß, 1996; Pelikan et al., 2006a).

It must be pointed out, however, that EDAs are not the only form way of incorporating learning in the optimization process. There are some approaches that perform this task by providing hybrid evolutionary/machine learning method, like, for example, the *learnable evolution model* (LEM) (Michalski, 2000). This approach is similar in spirit to EDAs although it also incorporates some EA features. Something that complicates its application in real-world practice. Furthermore, these efforts seem to have been concentrated on single-objective optimization (c. f. (Sheri and Corne, 2008, 2010)).

2.9 Estimation of Distribution Algorithms

EDAs have been claimed as a paradigm shift in the field of evolutionary computation. Like EAs, EDAs are population-based optimization algorithms. However, in EDAs the step where the evolutionary operators are applied to the population is substituted by construction of a statistical model of the most promising subset of the population. This model is then sampled to produce new individuals that are merged with the original population following a given substitution policy. Therefore, a benefit of EDAs is that not only do they return a solution to a problem, but a model representing the solutions is presented as well.

Because of this model-building feature EDAs have also been called *probabilistic-model-building genetic algorithms* (PMBGAs) (Pelikan et al., 1999b). A framework similar to EDAs is proposed by the *iterated density estimation evolutionary algorithms* (IDEAs) (Bosman, 2003). Figure 2.3 summarizes the workflow of an EDA in schematic form.

Model-building processes have evolved, too. Early approaches assumed that the different features of the decision variables were independent. Subsequent methods started to deal with interactions among the decision variables, first in pair-wise fashion and later in a generalized manner, using n-ary dependencies.

Multi-objective EDAs (MOEDAs) (Pelikan et al., 2006b) are the extensions of EDAs to the multi-objective domain. Most MOEDAs consist of a modification of existing EDAs whose fitness assignment function is substituted by one taken from an existing MOEA.

MOEDAs can be grouped in terms of their model-building approach. We will now give a brief description of MOEDAs, as this discussion is essential for our analysis. Note, however, that a comprehensive survey of current MOEDAs is beyond our scope. Instead we will concentrate of EDAs and their extrapolation to the multi-objective domain, therefore enumerating only those algorithms of interest. Table 2.1 summarizes the properties of the MOEDAs here described.

There are two complementing EDA approaches for storing or representing the search individuals. One keeps a population for search individuals and, in every iteration model the most promising subset of such population and create new individuals. On the contrary,

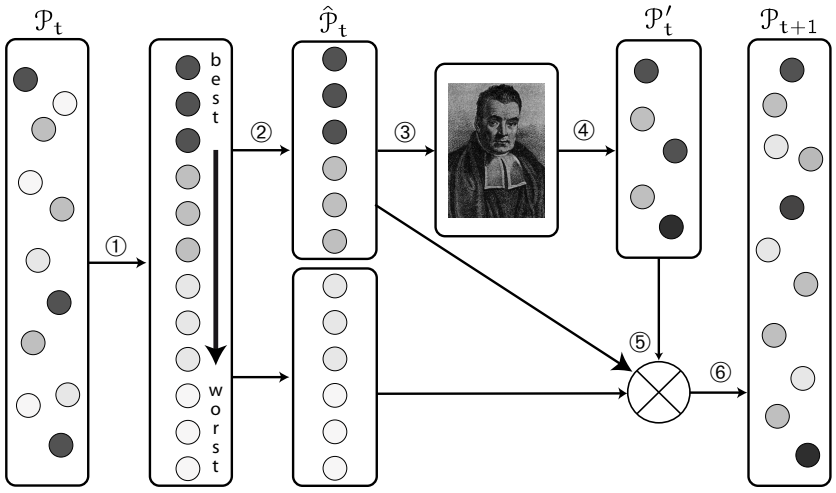


Figure 2.3: Diagram representation of an iteration of an estimation of distribution algorithm. Individuals of the current population, $\mathcal{P}_t$, have their fitness values calculated (1). The individuals are sorted with regard to their fitness (2) and a subset $\hat{\mathcal{P}}_t$ with the best elements are selected (3). $\hat{\mathcal{P}}_t$ is used as dataset for model building (4). This model is sampled to create new individuals, $\mathcal{P}'_t$, (5) that are combined with the current population to produce a new population (6), $\mathcal{P}_{t+1}$.

there are other approaches that store search information as the learned model. Therefore, this model is sampled and updated based on the adequacy of the sample. The first approach is, to the best of our knowledge, the most popular one in the multi-objective context.

2.9.1 Graphical Algorithm MOEDAs

One of the most common foundations for MOEDAs is a set of single-objective EDAs that build the population model using graphical models (Bishop, 2006). Most single-objective EDAs in that class rely on Bayesian networks (Pearl, 1988). This is the case of the Bayesian optimization algorithm (BOA) (Pelikan et al., 1999a), the estimation of Bayesian network algorithm (EBNA) (Etxeberria and Larrañaga, 1999) and the learning factorized distribution algorithm (LFDA) (Mühlenbein and Mahnig, 1999). Of these, BOA was the algorithm extrapolated to the multi-objective domain.

A Bayesian network is a probabilistic graphical model that represents a set of variables and their probabilistic (in)dependencies. They are directed acyclic graphs whose nodes represent variables, and whose arcs encode conditional independencies between the variables. Nodes can represent any kind of variable; either a measured parameter, a latent variable or a hypothesis.

The exhaustive synthesis of a Bayesian network from the algorithm's population is an *NP*-hard problem (Cooper, 1990; Dagum and Luby, 1993). Therefore, the intention behind the former approaches is to provide heuristics for building a network of reasonable computational complexity. BOA uses the so-called K2 metric, based on the Bayesian Dirichlet metric (Cooper and Herskovits, 1992), to assess the quality of a network. A simple greedy algorithm is used to add edges in each iteration.

BOA-based MOEDAs combine the Bayesian model-building scheme with an already ex-

isting Pareto-based fitness assignment. This is the case of the multi-objective BOA (mBOA) (Khan et al., 2002) that exploits the fitness assignment used in NSGA-II. Another algorithm based on hierarchical BOA (hBOA) (Pelikan, 2005a; Pelikan and Goldberg, 2006; Pelikan et al., 2000), called mhBOA (Khan, 2003; Pelikan et al., 2005), also uses the same form of fitness assignment but introduces clustering in the objective function space. A similar idea is proposed in (Laumanns and Ocenasek, 2002; Ocenasek, 2002), where the mixed BOA (mBOA) (Ocenasek and Schwarz, 2002) is combined with the SPEA2 selection scheme to form the multi-objective mBOA (mmBOA).

The multi-objective real-coded BOA (MrBOA) (Ahn, 2006) also extends a preexisting EDA, namely, the real-coded BOA (rBOA) (Ahn et al., 2004). RBOA performs a proper problem decomposition by means of a Bayesian factorization and probabilistic building-block crossover by employing mixture models at the level of subproblems. MrBOA combines the fitness assignment of NSGA-II with rBOA.

For the following experiments we followed the model-building strategy used by rBOA (Ahn et al., 2004), that is, apply a simple incremental greedy approach to construct the network. It adds edges to an initially fully disconnected graph. Each edge is added in order to improve, at each step, a particular formulation of the Bayesian information criterion (BIC) (Schwarz, 1978). Then, the conditional probabilities that take part of the Bayesian factorization are computed for each disconnected subgraph.

Note, finally, that Bayesian networks are not the only graphical model suitable for model-building. Other approaches, in particular Markov random fields (Kindermann and Snell, 1980), have also been applied in single-objective EDAs (Santana, 2003, 2005; Shakya and McCall, 2007). To the best of our knowledge, however, these approaches have not yet been extended to multi-objective problems.

2.9.2 Mixture Distribution MOEDAs

Another approach to modeling the subset with the best population elements is to apply a distribution mixture approach. In a series of papers, Bosman and Thierens (Bosman and Thierens, 2002, 2003, 2005; Thierens, 2003; Thierens and Bosman, 2001a,b) proposed several variants of their *multi-objective mixture-based iterated density estimation algorithm* (MIDEA). They are based on their IDEA framework. Bosman and Thierens proposed a novel Pareto-based and diversity-preserving fitness assignment function. The model construction is inherited from the single-objective version. The proposed MIDEAs considered several types of probabilistic models for both discrete and continuous problems. A mixture of univariate distributions and a mixture of tree distributions were used for discrete variables. A mixture of univariate Gaussian models and a mixture of multivariate Gaussian factorizations were applied for continuous variables. An adaptive clustering method was used to determine the capacity required to model a population.

MIDEAs do not place any constraints on the location of the centers of the distributions. Consequently, the MIDEA clustering mechanism does not provide a specific mechanism to ensure equal coverage of the Pareto-optimal front if the number of representatives in some parts of the front is much larger than the number of representatives in some other parts.

The clustering algorithms applied for this task include the randomized leader algorithm

(Hartigan, 1975), the *k*-means algorithm (MacQueen, 1967) and the expectation maximization algorithm (Dempster et al., 1977).

The *leader algorithm* (Hartigan, 1975) is a fast and simple partitioning algorithm that was first used in the EDA context as part of the IDEA framework. Its use is particularly appropriate in situations where the overhead introduced by the clustering algorithm must remain as low as possible. Besides its small computational footprint, this algorithm has the additional advantage of not having to explicitly specify in advance how many partitions should be discovered. On the other hand, the drawbacks of the leader algorithm are that it is very sensitive to the ordering of the samples and that the values of its thresholds must be guessed *a priori* and are problem dependent.

The algorithm goes over the data set exactly once. The distances from each sample to each of the cluster centroids are determined. Then, the cluster whose distance is smallest and below a given distance threshold, ρ_{Ld}, is selected. If no such cluster can be found, a new one is created, containing just this sample. Once the number of samples in a cluster has exceeded the sample count threshold ρ_{Lc}, the leader is substituted by the mean of the cluster members. The mean of a cluster changes whenever a sample is added to that cluster. After clustering, a Gaussian mixture is constructed, as described for the *naïve MIDEA* (Bosman and Thierens, 2005). This way the model can be sampled in order to produce new elements.

The *k-means algorithm* (MacQueen, 1967) is a well-known machine learning method. It constructs k partitions of the input space. To do this, it uses partition centroids. First, the k centroids are initialized from randomly selected samples. At each iteration, each sample is assigned to the nearest partition based on the distance to the partition centroid. Once all of the points have been assigned, the means of the partitions are updated. The algorithm iterates until the centroids no longer change significantly. An important issue in this algorithm is how to set parameter k such that partitioning is adequate. Parameter setting requires some experience. In the context of MIDEAs (Bosman, 2003) the approach followed is to increment k and calculate the negative log-likelihood of the mixture probability distribution after estimating a factorized probability distribution in each cluster. If the resulting mixture probability distribution is significantly better than for a smaller value of k, this value is accepted and the search continues. As in the previous case, after the clusters are determined, a Gaussian mixture is estimated for sampling purposes.

The *expectation maximization (EM) algorithm* (Dempster et al., 1977) is an iterative approach to computing a maximum likelihood estimate. EM uses the difference in the negative log-likelihood of the estimated probability distribution between subsequent iterations in order to derive the hidden parameters. In a clustering context, EM is used to get an approximation of the maximum likelihood estimation of a mixture probability distribution. The number of components in the mixture probability distribution is usually chosen beforehand. This choice is similar to the choice of the number of partitions when using a clustering approach to the estimation of a mixture probability distribution from data. In this case a similar approach to the one discussed for *k*-means is applied.

MIDEAs are not the only mixture-based algorithms. The *multi-objective Parzen EDA* (MOPED) (Costa and Minisci, 2003; Costa et al., 2003) puts forward a similar mixture-based approach. MOPED uses the NSGA-II ranking method and the Parzen estimator (Parzen, 1962) to approximate the probability density of solutions lying on the Pareto front. The proposed algorithm has been applied to different types of test case problems, and results

Table 2.1: Summary of the main characteristics of the different MOEDAs discussed.

MOEDA	Domain	Fitness Assignment	Model-Building	Original EDA
mBOA	combinatorial	NSGA-II	Bayesian	BOA
mhBOA	combinatorial	NSGA-II	Bayesian	hierarchical BOA
mmBOA	comb. + cont.	SPEA2	Bayesian	mixed BOA
MrBOA	continuous	NSGA-II	Bayesian	real-coded BOA
Naïve MIDEA	continuous	Proprietary	Univariate dists.	IDEA
MO-CMA-ES	continuous	Indicator	Covariance matrix	CMA-ES
RM-MEDA	continuous	NSGA-II	Regularity prop.	–
MOPED	continuous	NSGA-II	Parzen estimator	–

show a good performance of the overall optimization procedure in terms of the total number of objective function evaluations.

2.9.3 Covariance Matrix Adaptation Evolution Strategies

Covariance matrix adaptation evolution strategies (CMA-ES) (Hansen et al., 2003; Hansen and Ostermeier, 2001) have been shown to yield many outstanding results in comparative studies (Auger, 2009a,b; Auger and Hansen, 2009). CMA-ES consists of a method for updating the covariance matrix of the multivariate normal mutation distribution used in an evolution strategy (Beyer and Schwefel, 2002). They can be viewed as an EDA, as new individuals are sampled according to the mutation distribution. The covariance matrix describes the pairwise dependencies between the variables in the distribution. Adaptation of the covariance matrix is equivalent to learning a second-order model of the underlying objective function. CMA-ES has been extrapolated to the multi-objective domain (Igel et al., 2007) by using an hypervolume-based selection.

2.9.4 Other Approaches

Other MOEDAs have been proposed in order to take advantage of the mathematical properties of the Pareto-optimal front. For example, the *regularity model-based multi-objective estimation of distribution algorithm* (RM-MEDA) (Zhang et al., 2008; Zhou et al., 2005) is based on the regularity property derived from the Karush–Kuhn–Tucker condition. This means that, subject to certain constraints, the Pareto-optimal set, $\mathcal{D}^*$, of a continuous multi-objective optimization problem can be induced to be a piecewise continuous $(M-1)$-dimensional manifold, where M is the number of objectives (Miettinen, 1999; Schütze et al., 2003).

At each iteration, RM-MEDA models the promising area of the decision space using a probability distribution whose centroid is a $(M-1)$–dimensional piecewise continuous manifold. The local principal component analysis algorithm (Kambhatla and Leen, 1997) is used to build this model. New trial solutions are sampled from the model thus built. Again, this model adopts the fitness assignment mechanism proposed by NSGA-II. The main drawback of this algorithm is its high computational complexity. This is an obstacle to its application in problems with many objective functions.

2.10 The Model Building Issue

An analysis of the results yielded by current multi-objective EDAs and their scalability against the number of objectives leads to the identification of some issues that could be preventing MOEDAs from getting substantially better results than other evolutionary approaches. Such issues include:

1. incorrect treatment of data outliers;
2. loss of population diversity; and
3. excess of computational effort devoted to finding an optimal population model.

These issues can be traced back to the single-objective predecessor of most MOEDAs and its respective model-building algorithms.

Improving the model-building algorithm seems to be a promising direction for research as; to the best of our knowledge, it has not been properly addressed. So far, MOEDA approaches have mostly used off-the-shelf machine learning methods. However, the task in discussion has different characteristics than those for which those methods were originally meant.

As the model-building feature is the essential difference between MOEDAs and MOEAs, MOEDA underperformance can me mostly attributed to it. In the next chapter the nature of the model building issue in depth and we will look for solutions to this matter.

2.11 Convergence and Stopping Criteria

Most soft-computing, heuristic, non-deterministic or numerical methods all have in common that they need a stopping criterion. The stopping criterion, which is usually a heuristic itself, is responsible for minimizing the wastage of computational resources by detecting scenarios where it makes no sense to continue executing the method.

The success or failure of any practical application relies heavily on not only the techniques applied but also the support methodologies, including the stopping criterion. Paradoxically, this is a matter that has often been overlooked by the community, probably because it plays a supporting part. This relegates the issue to an apparently secondary role. Consequently, the theoretical and practical implications concerning this topic have not yet been properly explored. Indeed, many real-world applications of theoretically outstanding methods may have underperformed due to an incorrect algorithm termination scheme.

Typically, the stopping criterion is invoked at the end of an iteration of the algorithm. At that point, it is decided whether algorithm execution should continue or can be aborted. We have identified four scenarios when the execution of an algorithm should terminate:

1. the current solution is satisfactory;
2. the method is able to output a feasible solution, which, although not optimal, is unlikely to be bettered;

3. the method is unable to converge to any solution, or
4. the computation already performed is sufficient to reach a solution or further computation is unjustified.

Besides detecting the situations in which the algorithm should be stopped, a stopping criterion should be as lightweight as possible in terms of computational complexity. Computing resources have to be expended on the algorithm itself, and, if the criterion is burdensome, it is more likely to be a weakness than a plus point.

Stopping criteria can be grouped into local (iteration-wise) criteria and global (execution-wise) criteria. Local criteria have access only to data pertaining to each iteration of the method. They measure the difference between the current solution and a predefined reference or optimal value and then decide when they are close enough. This type of criterion has the obvious and paradoxical shortcoming of requiring a priori knowledge of the desired optimal value of the solution. This potential weakness has no significant impact if the class of problem being addressed allows the reference value to be replaced by the axis "zero" reference. This applies, for example, to function approximation and other types of problems that can be reduced to an error minimization problem.

On the other hand, global criteria keep track of the process progress across different iterations in order to make decisions relying on the long-term behavior of the algorithm being monitored. This evidence-gathering process has two positive impacts: (i) algorithm progress can be assessed in a relative fashion by comparing the outcome of different iterations and (ii) algorithm progress is more resilient to local optima and noise as it takes into account different iterations.

Similarly, these criteria can be classified in online and offline stopping criteria. Online criteria are used to diagnose the progress of the algorithm as the execution proceeds while offline criteria are applied to analyze the results of a given algorithm to determine at which iteration those results are most valuable. Online stopping criteria are generally considered more interesting from a practical point of view.

The simplest approach to stopping is to compute how well the current the algorithm state satisfies a given quality threshold. These local (or iteration-wise) criteria only exploit information present in the context of the iteration. Therefore, their analysis horizon is limited, and it is impossible for them to assess the progress of the algorithm across consecutive iterations.

The solution to this is to employ a *evidence gathering process* (EGP) that gathers evidence of progress measured across iterations. This class of global or execution-wise scheme can apply a statistical or machine learning approach to combine the different measurements.

Evolutionary algorithms also require a stopping criterion, but the vast majority of applications have bypassed this matter by using a termination scheme that specifies a finite number of iterations. Furthermore, the research results that has addressed this issue (Hernandez et al., 2005; Safe et al., 2004; Zielinski and Laur, 2007; Zielinski et al., 2005) does not appear to have propagated to the rest of the research community.

This is especially applicable to MOEAs. In the multi-objective case, a local criterion must measure the similarity between the current and the Pareto-optimal front and decide when they are close enough (Zitzler et al., 2002b). This type of criterion has the obvious paradoxical shortcoming of requiring an a priori known Pareto-optimal front.

On the other hand, global approaches may apply relative improvement metrics that analyze the partial results of the algorithm across iterations. Therefore, there is no need to resort to an absolute comparison with an a priori established threshold. In the particular case of MOEAs, this type of criterion should compare the non-dominated solution fronts yielded by different iterations in order to determine how the optimization process is progressing.

There are a number of quality indicators (see Section 2.2) that can be repurposed for this task as *progress indicators* (PIs), but their high computational cost is an obstacle to their application. While there has been little theoretical research dealing with MOEA convergence (Hanne, 1999; Rudolph and Agapie, 2000), there have been even fewer attempts to deal with the stopping issue.

The status of this issue has recently started to change, as interest in these matters has grown. This can be inferred from the relatively large body of research that has gradually started to deal with this question, as we will now show.

2.11.1 Formal Description

[2]

An *online stopping criterion* (OSC) can be formally defined as a 4-tuple,

Definition 2.15 (online stopping criterion)

$$
\begin{array}{lll}
OSC := & \{\mathcal{S}, \Pi(\cdot), \Upsilon(\cdot), \Phi(\cdot)\}, & \text{where,} \\
& \mathcal{S} : \textit{data structure}, & \textit{stores the internal state of the criterion,} \\
& \Pi : \mathcal{PF}_t^* \times \mathcal{S} \to \mathcal{S}, & \textit{progress indicators (PIs),} \\
& \Upsilon : \mathcal{S} \to \mathcal{S}, & \textit{evidence gathering process (EGP),} \\
& \Phi : \mathcal{S} \to \{\text{true}, \text{false}\}, & \textit{stop decision function.}
\end{array}
\tag{2.22}
$$

In the state $\mathcal{S}$, all information required for the computations of the EGP are stored. It necessarily includes the input data $\mathcal{M}$ for the EGP. In the following, we use $\mathcal{S}.\mathcal{M}$ in order to address the current version of $\mathcal{M}$ stored in the state $\mathcal{S}$. The state $\mathcal{S}$ can additionally contain previous Pareto front approximations or PI values, an external archive, or flags indicating whether the threshold has been reached in the last generations. These information can be changed or used in different functions of the taxonomy and are therefore exchanged via $\mathcal{S}$. The data stored in the state ensures that the OSC can make the stopping decision just based on the Pareto front approximation $\mathcal{PF}_t^*$ of the current generation.

The function $\Pi : \mathcal{PF}_t^* \times \mathcal{S} \to \mathcal{S}$ uses the PIs to update the input data $\mathcal{S}.\mathcal{M}$ for the EGP. This general type of function is introduced since the update can differ depending on the considered PIs, e. g., some approaches update the PI of all preceding generations based on the current generation $\mathcal{PF}_t^*$, whereas others only update the values of the last generation. Consequently, the size of $\mathcal{S}.\mathcal{M}$ can be up to $P \times t_{\text{mem}}$, where P is the number of PIs and t_{mem} is the number of preceding generations considered in the EGP. In Π are also performed all state updates required for the PI computation, such as the update of the archive and the storage of previously computed PI values. Consequently, the input data $\mathcal{S}.\mathcal{M}$ is a necessary

[2] This section is based on the survey by Wagner et al. (2011). The other authors of the paper have made a substantial contribution to it.

part of the updated state, as it would restrict the generality of the framework as sole output of Π.

The function $\Upsilon : \mathcal{S} \rightarrow \mathcal{S}$ encodes the EGP. It updates the state of the criterion based on the input data $\mathcal{S}.\mathcal{M}$ included in the current state. Usually, the EGP returns one aggregated value per PI, but also a combined analysis like in OCD (Wagner et al., 2009) can be performed. In this case, the EGP value of the combined analysis is assigned to all considered PI.

The decision function $\Phi : \mathcal{S} \rightarrow \{\text{true}, \text{false}\}$ finally determines whether the current state of the criterion indicates that the expected improvement of the MOEA is below the predefined threshold ε, i. e., the MOEA should be stopped. For this decision, the EGP value, but also additional information, such as the estimation error and the degrees of freedom in the estimation of the EGP value, are usually utilized. The decision function can only return a single Boolean. If multiple EGPs are considered in parallel, also the aggregation of the corresponding decisions has to be performed in Φ.

Using these formalisms, the procedure of a generic OSC can be implemented as shown in Figure 2.4. The user has to specify the MOEA, the problem of interest and the maximum affordable number of generations, $t_{\max}$, as well as PI-related data of the problem, such as a reference set and the ideal and nadir points (Deb et al., 2010). The actual OSC is specified by the combination of the PIs, the EGPs, and the stopping decisions. For each step, also multiple functions can be provided.

After the initialization of the state in which the archive is initialized and information about the chosen PI and EGP are stored, the control parameters of the OSC are initialized. After each generation of the MOEA, $\mathcal{S}.\mathcal{M}$ and the required data structures are updated using the chosen Π_i. If there are t_{mem} measurements, the functions $\Upsilon_j(\cdot)$ are applied in order to attach the EGP value for each PI to $\mathcal{S}$. Finally, $\Phi_k(\cdot)$ can be applied to determine whether the algorithm should be stopped.

2.11.2 A Survey on Stopping Criteria

We now present a survey of the state-of-the-art OSC in chronological publication date order. These approaches are described using the proposed formalization presented on Definition 2.15. A summary is provided in Table 2.2. It must be pointed out that this section is based on the forthcoming survey by Wagner et al. (2011). The approaches discussed are part of a taxonomy-based framework that can be downloaded from the author's web site (Wagner and Martí, 2010).

Deb and Jain (2002): Running Metrics

Deb and Jain (2002) were the first authors who proposed the investigation of performance metrics over the run of the MOEA. They used two metrics, one for evaluating the convergence and one for measuring the diversity of $\mathcal{PF}_t^*$. The *convergence metric* (CM) calculates the average of the smallest normalized Euclidean distance from each individual in $\mathcal{PF}_t^*$ to a precomputed reference set. For the computation of the *diversity metric* (DVM), all objective vectors of $\mathcal{PF}_t^*$ are projected onto a hyperplane of dimension $m-1$ which is then uniformly divided into discrete grid cells.

Parameters:
▷ Multi-objective evolutionary algorithm of interest.
▷ Multi-objective problem of interest.
▷ $t_{\max}$, maximum number of iterations.
▷ $\mathcal{PI}$, set of PI functions Π_i.
▷ $\mathcal{EGP}$, set of EGP functions Υ_j.
▷ $\mathcal{SDF}$, set of stopping decision functions Φ_k, $k = \{1, \ldots, K\}$.
▷ Problem-based parameters (reference set, ideal and nadir points).
▷ Manually defined settings of control parameters (optional).
Initialize state $\mathcal{S}$.
Initialize control parameters of Π_i, Υ_j, and Φ_k.
$t = 0$.
while $t < t_{\max}$ **do**
  $t = t + 1$.
  Perform one generation of the MOEA and obtain $\mathcal{PF}^*_t$.
  for each indicator Π_i in $\mathcal{PI}$ **do**
    Update input data $\mathcal{S}.\mathcal{M}$ and PI-dependent information, $\mathcal{S} = \Pi_i(\mathcal{PF}^*_t, \mathcal{S})$.
  end for
  if $|\mathcal{S}.\mathcal{M}| = t_{\text{mem}}$ **then**
    for each EGP Υ_j in $\mathcal{EGP}$ **do**
      Update EGP value based on $\mathcal{S}.\mathcal{M}$, $\mathcal{S} = \Upsilon_j(\mathcal{S})$.
    end for
    for each stopping decision function Φ_k in $\mathcal{SDF}$ **do**
      Compute stop decision, stop(k) = $\Phi_k(\mathcal{S})$
    end for
    if $\forall k$: stop(k) = true **then**
      Stop MOEA!
      return t and $\mathcal{S}$.
    end if
  end if
end while

Figure 2.4: Algorithmic representation of a general online stopping criterion conforming Definition 2.15.

DVM tracks the number of attained grid cells and also evaluates the distribution by assigning different scores for predefined neighborhood patterns. In order to avoid bad DVM values based on unattainable grid cells, again a reference set is used. The EGP and the final decision then rely on a visual inspection of the progression of the CM and DVM by the user. Consequently, the state $\mathcal{S}$ of this criterion contains the reference set and all values of the CM and DVM computed until the current generation.

Rudenko and Schoenauer (2004): Stability Measure

Rudenko and Schoenauer (2004) defined a *stability measure* for the $\mathcal{PF}^*_t$ of NSGA-II (**?**). Their experimental studies showed that the stagnation of the maximum crowding distance (maxCD) within $\mathcal{PF}^*_t$ is a suitable indicator for NSGA-II convergence. Thus, the standard

deviation of the last t_{mem} values of the maximum crowding distance is used as EGP (STD).

For the computation, the last $t_{\text{mem}} - 1$ values of $maxCD$ are contained in the state $\mathcal{S}$. In each generation, $\mathcal{S}$ is updated using the current $maxCD$ value and STD is computed. The decision step requires a user defined threshold ε leading to an NSGA-II termination once the STD falls below this value (Threshold).

Martí et al. (2007, 2009a): MGBM Criterion

Martí et al. (2007, 2009a) proposed the *MGBM stopping criterion* (according to the authors' last names), which combines the *mutual domination rate indicator* (MDR) with a simplified Kalman filter that is used as EGP. The function Π considers $\mathcal{PF}^*_{t-1}$ and $\mathcal{PF}^*_t$ and applies the MDR indicator to update $\mathcal{S}.\mathcal{M}$. Thus, the Pareto front of the previous generation has to be stored in the state $\mathcal{S}$.

The EGP function Υ applies the Kalman filter and updates the Kalman state and the corresponding estimated error in $\mathcal{S}$. The decision function Φ is realized by stopping the MOEA when the confidence interval of the a-posteriori estimation completely falls below the prespecified threshold ε.

This criterion is put forward as part of this thesis. Further details are provided in Chapter 5.

Wagner et al. (2009): Online Convergence Detection (OCD)

In the OCD approach, the established performance measures hypervolume, R2 and additive epsilon indicator are used as PIs. The function Π updates all t_{mem} PI values stored in $\mathcal{S}.\mathcal{M}$ using the current generation $\mathcal{PF}^*_t$ as reference set. Consequently, the sets $\mathcal{PF}^*_{t-t_{\text{mem}}}$ to $\mathcal{PF}^*_{t-1}$ have to be additionally stored in the state $\mathcal{S}$. In Υ, the variance of the values in $\mathcal{S}.\mathcal{M}$ is computed for each PI. Moreover, a least-squares fit of a linear model with slope parameter β is performed based on the individually standardized values in $\mathcal{S}.\mathcal{M}$.

In Φ, the variance is then compared to a threshold variance ε by means of the one-sided χ^2-variance test with H_0: $\text{VAR}(\mathcal{S}.\mathcal{M}) \geq \varepsilon$ and a p-value is looked up. By testing the hypothesis H_0: $\beta = 0$ by means of a t-test, a second p-value is obtained. For these tests, the variance obtained by STD, β, and its standard error have to be stored in the state. The same holds for the resulting p-values.

The MOEA is stopped when the p-values of two consecutive generations are below the critical level $\alpha = 0.05$ for one of the variance tests (the null hypothesis H_0 is rejected) or above $\alpha = 0.05$ for the regression test (H_0 is accepted). Consequently, the p-values of the preceding generations have to be stored in $\mathcal{S}$.

In (Wagner and Trautmann, 2010) a reduced variant of the OCD approach for indicator-based MOEA was introduced. This approach was illustrated for the hypervolume indicator and the SMS-EMOA (Beume et al., 2007) (OCD-HV). Since the hypervolume is a unary indicator, only the absolute hypervolume values have to be stored. The previous $\mathcal{PF}^*_t$ can be neglected. For better compliance with the other PI, the differences to the value of the current set $\mathcal{PF}^*_t$ are stored in $\mathcal{S}.\mathcal{M}$ in order to minimize the PI. In case the internally optimized performance indicator monotonically increases, as for the SMS-EMOA and the hypervolume,

OCD should only consider this PI. The regression test can be neglected. Consequently, the complexity of OCD is reduced by concentrating on the variance test for one specific PI.

Bui et al. (2009): Dominance-Based Quality of $\mathcal{P}$ (DQP)

Bui et al. (2009) introduce a dominance-based stability measure which approximately evaluates the local optimality of a solution (DQP). The DQP is the only PI that requires many additional evaluations of the objective function for estimating the ratio of dominating solutions in the neighborhood of a solution. A Monte Carlo simulation with 500 evaluations per solution in $\mathcal{PF}_t^*$ was used. Consequently, the DQP is a very expensive, but powerful measure. No additional state information or EGPs are required. No clear guidelines for stopping the MOEA are provided. Instead, a visual analysis of the convergence behavior and possible stagnation phases is performed. However, a clear stopping criterion would be $DQP = 0$, as this would be the case when no local improvements are possible. In fact DQP is closely related to the gradient of a solution in single-objective optimization. In line with this observation, the authors also use DQP as measure for guiding a local search (Bui et al., 2009).

Guerrero et al. (2010): Least Squares Stopping Criterion (LSSC)

LSSC (Guerrero et al., 2010) can be seen as an approach to integrate both EGP of OCD into a single EGP and to also simplify the PI computation and the stopping decision. Therefore, only one PI is considered and the variance-based EGP and the statistical tests for the stopping decision are omitted. Still, a regression analysis of the PI is performed as EGP and the PI values of the last t_{mem} generations are updated using the current generation as reference set. Thus, the last t_{mem} Pareto front approximations have to be stored in the state $\mathcal{S}$ in order to update $\mathcal{S}.\mathcal{M}$.

In contrast, the PIs are not standardized allowing the estimation of the expected improvement by means of the slope β. If β falls below the predefined threshold ε, the MOEA is stopped. In order to prevent a loss of robustness by omitting the statistical tests, a threshold for a goodness-of-fit test based on the regression residuals is computed via the Chebyshev inequality. Only if the model is valid, the estimated slope is compared to ε. Consequently, the analyses performed in OCD and LSSC differ.

Whereas LSSC directly tries to detect whether the expected improvement falls below the allowed threshold ε, OCD tests the significance of the linear trend whereas the magnitude of the expected improvement is evaluated via the variance of $\mathcal{S}.\mathcal{M}$.

Goel and Stander (2010): Non-dominance-based Convergence Metric

Goel and Stander (2010) use a dominance-based PI based on an external archive of non-dominated solutions which is updated in each generation. The current archive is stored in $\mathcal{S}$ and is used to determine the CR. The authors provide empirical evidence for the robustness of the CR, so that no EGP is applied (Direct). The stopping decision is made by comparing the CR with a predefined threshold ε (Threshold).

In addition, an utility-based approach is proposed. The utility is defined as the difference in the CR between the generations t and $t - t_{\text{mem}}$. In order to increase the robustness of the approach, a moving average $U_t^* = (U_t + U_{t-t_{\text{mem}}})/2$ is used as EGP (Moving). The MOEA is stopped when the utility falls below an adaptively computed threshold $\varepsilon_{\text{adaptive}}$. Moreover, a minimum CR of $\text{CR}_{\text{min}} = 0.5$ has to be reached in order to avoid a premature stopping due to perturbances in early generations. The adaptive threshold $\varepsilon_{\text{adaptive}}$ is defined as the fraction $\text{CR}_{\text{init}}/(F \cdot t_{\text{init}})$ of the initial utility U_{init}, which corresponds to the first CR value CR_{init} exceeding 0.5 and the corresponding generation t_{init}. F is a user parameter that specifies which ratio of the averaged initial utility $\text{CR}_{\text{init}}/t_{\text{init}}$ is at least acceptable. For this version, also $\varepsilon_{\text{adaptive}}$ and $\text{U}^*_{t-t_{\text{mem}}}$ have to be stored in $\mathcal{S}$.

Table 2.2: Summary of the stopping criteria survey organized by components of the 4-tuple of Definition 2.15. Acronyms and abbreviations are introduced in Section 2.11.

Approach	**PIs,** $\Pi(\mathcal{PF}_t^*, \mathcal{S})$	**Internal state,** $\mathcal{S}$	**EGP,** $\Upsilon(\mathcal{S})$	**Stop desc.,** $\Phi(\mathcal{S})$	**Parameters (defaults)**
Running metrics (Deb and Jain, 2002)	CM, DVM $\lvert\mathcal{S}.\mathcal{M}\rvert = 2 \times 1$	– $\mathcal{M}$: All CM and DVM values – Reference set	Attach current CM and DVM values to state	Visual check by decision maker	none
Stability measure (Rudenko and Schoenauer, 2004)	maxCD $\lvert\mathcal{S}.\mathcal{M}\rvert = 1 \times 1$	– $\mathcal{M}$: maxCD of gen. $(t - t_{\text{mem}})$ to t – STD	STD of $\mathcal{S}.\mathcal{M}$	Threshold	– t_{mem} (40) – ε (0.02) – hits h (1)
MBGM (Martí et al., 2007, 2009a)	MDR $\lvert\mathcal{S}.\mathcal{M}\rvert = 1 \times 1$	– $\mathcal{M}$: Current MDR value – $\mathcal{PF}_{t-1}^*$ – Kalman state with corresp. STD	Kalman	CInormal	– ε (0) – p-CI (97.725 %) – Kalman inertia R (0.1) – hits h (1)
OCD-Classic (Naujoks and Trautmann, 2009; Wagner and Trautmann, 2010; Wagner et al., 2009)	Eps, R, HV (parallel) $\lvert\mathcal{S}.\mathcal{M}\rvert = 3 \times t_{\text{mem}}$	– $\mathcal{M}$: Eps, R, and HV of gen. $(t - t_{\text{mem}})$ to $(t-1)$ – $\mathcal{PF}^*$ of gen. $(t - t_{\text{mem}})$ to $(t-1)$ – Current slope β with corresp. STD – Current STD – p-values of gens. t and $(t-1)$	– STD of each PI in $\mathcal{S}.\mathcal{M}$ – Reg on $\mathcal{S}.\mathcal{M}$ (individually standardized)	– χ^2-test: STD $< \varepsilon$ – t-test: $\beta = 0$	– t_{mem} (16) – ε (0.001) – hits h (2)
DQP (Bui et al., 2009)	DQP $\lvert\mathcal{S}.\mathcal{M}\rvert = 1 \times 1$	$\mathcal{M}$: Current DQP value	Direct	Visual check by decision maker	– samples N (500) – radius r (0.05)
LSSC (Guerrero et al., 2010)	Eps, HV, MDR (separately) $\lvert\mathcal{S}.\mathcal{M}\rvert = 1 \times t_{\text{mem}}$	– $\mathcal{M}$: Eps, HV, or MDR of gen. $(t - t_{\text{mem}})$ to $(t-1)$ – $\mathcal{PF}^*$ of gen. $(t - t_{\text{mem}})$ to $(t-1)$ – Current slope β	Reg on each PI in $\mathcal{S}.\mathcal{M}$	validThreshold	– t_{mem} (30) – ε (HV: 0.002, Epsilon: 0.0004, MDR: 0.00002) – hits h (1)
OCD-HV (Wagner and Trautmann, 2010)	HV $\lvert\mathcal{S}.\mathcal{M}\rvert = 1 \times t_{\text{mem}}$	– $\mathcal{M}$: Differences between the HVs of gen. $(t - t_{\text{mem}})$ to $(t-1)$ and t – p-values of gens. t and $(t-1)$	STD of $\mathcal{S}.\mathcal{M}$	– χ^2-test: STD $< \varepsilon$	– t_{mem} (14) – ε (0.0001) – hits h (2)
CR (Goel and Stander, 2010)	CR $\lvert\mathcal{S}.\mathcal{M}\rvert = 1 \times 1$	– $\mathcal{M}$: CR of gen. $(t - t_{\text{mem}})$ to t – Archive of non-dominated ind. – $\varepsilon_{adaptive}$ – $\text{U}^*_{t-t_{mem}}$	Direct or Moving average	CR $> \varepsilon$ or $\text{U}^*_t < \varepsilon_{adaptive}$	– t_{mem} (10) – ε (0.8) – utility ratio F (10) – hits h (1)

2.11.3 Remarks on Current Stopping Criteria

Basically, the existing PIs can be classified with respect to their optimization goal. One class is formed by the PIs based on analyzing the dominance relation between the current population (or archive) and a previous one, e. g., MDR and CR. Other approaches provide information about the distribution (maxCD, DVM) or local optimality of the solutions (DQP). Only a few of the PI try to combine some of these goals, e. g., hypervolume, R2, Epsilon, and CM, each with different trade-offs.

The dominance-based PI the convergence of the population to be formally assessed. The probability of improving the diversity and distribution and therewith the quality of the discrete approximation of O^* is not specifically addressed. The improvements in these PI will therefore reduce much faster. Moreover, the magnitude of the improvement generated by a new non-dominated solution is not considered. This information would be important in order to evaluate an expected improvement. As shown in the last years (Wagner et al., 2007), the dominance relation has only a weak explanatory power for many-objective problems.

The dominance-based PI usually reuse the information provided by the selection method of the MOEA. Thus, they do not require expensive additional calculations. PIs like CM, R, and hypervolume have to be additionally computed in each MOEA generation, where especially the dominated hypervolume has a complexity which increases exponentially with the objective space dimension. Bui et al. (2009) even perform additional evaluations for convergence detection. In general, the use of additional computational time or evaluations should be kept below the effort of the alternative option of just allowing the MOEA to precede for an affordable number of additional generations.

In addition, reference and nadir points, as well as reference sets, can be required for some PIs, e. g., the reference set for the CM and DVM, the ideal and nadir point for R2, and the reference point for hypervolume. In contrast to mathematical test cases, this information is usually not existing for practical applications. Strategies to obtain this data have to be derived which could comprise preliminary algorithm runs, random sampling, or evaluations on a grid covering the whole search space. Based on approximations of the objective boundaries, the normalization of the PI to unit intervals is possible —an approach that is often recommended (Deb and Jain, 2002; Zitzler et al., 2003). However, even the normalization can lead to scalarization effects which make the specification of thresholds difficult (Wagner and Trautmann, 2010). For the dominance-based indicators, usually relative amounts are calculated, e. g., $-1 \leq MDR \leq 1$ or $0 \leq CR \leq 1$, which facilitate the definition of adequate threshold values. Nevertheless, the only reasonable threshold for these approaches is $\varepsilon = 0$ based on the above considerations.

Some methods do not use a distinct EGP. They rely on a single evaluation of the considered PI. Due to the stochastic nature of MOEAs, it is obvious that those approaches will not be as robust as alternative ones using an EGP gathering PIs over a time window. Moreover, the EGP-based approaches are usually flexible with respect to the kind of integrated PI. By means of a suitable PI, the performance aspects (e. g., convergence, distribution, spread) which are the most important for the optimization task at hand can be considered in the OSC. In this context, also the considered MOEA has an important role. Mathematical convergence can only be expected if the corresponding MOEA is based on this PI, e. g., the SMS-EMOA in combination with the hypervolume (Beume et al., 2010). Furthermore, most OSC are designed for separately using a single PI. As performance of a MOEA has different aspects (Deb

and Jain, 2002; Zitzler et al., 2003), it should be analyzed if the usage of PIs covering these aspects of the approximation quality could support an efficient OSC decision.

Another important OSC design issue is concerned with the choice of the stopping decision. Statistical tests or confidence intervals lend themselves to draw robust decisions from random variables monitored over time. However, in order to choose an adequate test or distribution, some assumptions on the behavior of the considered PI are necessary. As a first approach, Mersmann et al. (2010) analyze the distribution of the final hypervolume value of different MOEAs. Among other characteristics it is shown to be uni-modal in most cases. Consequently, the use of classical tests is possible, maybe based on additional transformations.

The parametrization of the OSC requires special attention as well. Parameters have to be carefully chosen in order to obtain the desired results with respect to the trade-off between runtime and approximation quality. For most approaches, no clear guidelines for setting up the required parameters are given or a visual analysis is suggested (Bui et al., 2009; Deb and Jain, 2002). In contrast, Wagner and Trautmann (2010) empirically derive guidelines for reasonably setting the OCD parameters t_{mem} and ε based on statistical design-of-experiment methods. The resulting parameter recommendations can be found in Table 2.2. For reasonable comparisons between the OSC, such kind of studies should also be performed for the other OSC. Furthermore, the problems and possibilities resulting from a combination of the methods with respect to the proposed PI, EGP, and stopping decisions should be a matter of future research. In this context, an analysis of the compatibility of the PI, EGP, and decision criteria would be of special interest.

3

Understanding Model Building

Don't bite my finger, look where it's pointing.
Warren S. McCulloh

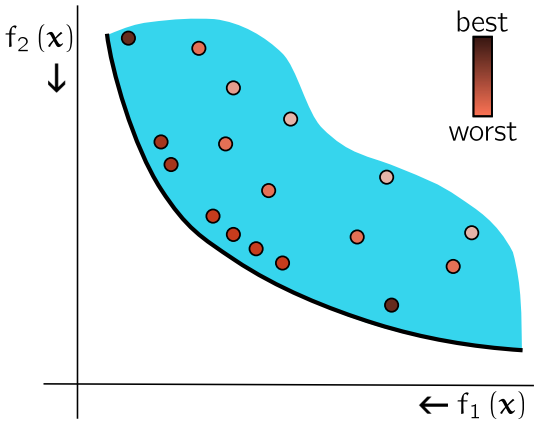
REGARDLESS of the many efforts aimed at providing usable model-building methods for EDAs, the nature of the problem itself has received relatively little attention. In spite of the succession of gradually improving results of EDAs, one question hangs over the search for possibilities for further improvement. Would current statistically sound and robust approaches be valid for the problem being addressed? Or, in other words, does the model-building problem have particular demands that can only be met by custom-made algorithms? Machine learning and statistical algorithms, although suitable for their original purpose, might not be that effective in the particular case of model building.

Generally, such algorithms are off-the-shelf machine learning methods that were originally intended for other classes of problems. On the other hand, the model-building problem has particular requirements that the above methods do not meet and may even go against. Furthermore, the consequences of this misunderstanding would be more dramatic when scaling up the number of objectives, since the situation is made worse by the implications of the curse of dimensionality (Bellman, 1961).

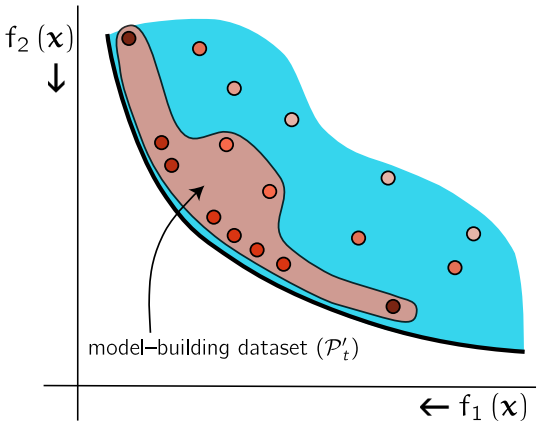
In this chapter we argue that the model-building problem has not been properly identified. For this reason, it has been treated like other previously existing problems overlooking that fact that this problem has particular requirements. This matter did not show up as clearly in single-objective EDAs. Thanks to the extension to the multi-objective domain this issue has become more evident, as we will debate in the remainder of this chapter.

3.1 Incorrect Treatment of Outliers

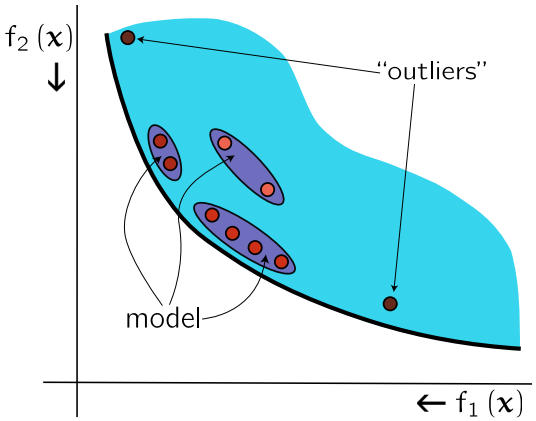
The isolated data elements or data outliers' issue is a good example of the deficient understanding of the nature of the model-building problem. In machine-learning practice, outliers are handled as noisy, inconsistent or irrelevant data. Therefore, outlying data is expected to have little influence on the model or it is just disregarded. However, this behavior is not appropriate for model-building. In this case, it is known beforehand that all elements in



(a) A population at a given iteration according to their Pareto-optimality and spread.



(b) Selection of model-building population subset with the best elements of the population.



(c) After model construction, isolated elements, which are the most relevant elements of the current population, are disregarded.

Figure 3.1: A graphical example of how standard model-building algorithms fail to take into account outliers.

the data set should be taken into account, as they represent newly discovered or candidate regions of the search space and, therefore, must be explored.

Therefore, these instances should be at least equally represented by the model and perhaps even reinforced. This situation is illustrated in Figure 3.1. A model-building algorithm that primes outliers might actually speed up the search process and lower the rate of the exponential dimension-population size dependency.

As model-building strategies varies from EDA to EDA, it is hard to back the previous discussion with a general theoretical support. In order to do so, we must define an individual z_i as the pair representing values in decision and objective sets,

$$z_i = \langle x_i, F(x_i) \rangle \ . \tag{3.1}$$

In a simplified case, we can state that model building is an unsupervised machine learning

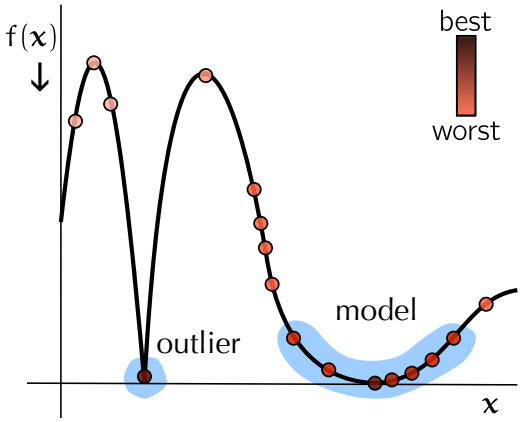


Figure 3.2: An example of how model-building algorithms might fail to take into account outliers in sigle-objective multi-modal optimization.

problem with learning dataset,

$$\Psi = \{\boldsymbol{x}_i\}\,; \forall \boldsymbol{z}_i = \langle \boldsymbol{x}_i, \boldsymbol{F}(\boldsymbol{x}_i)\rangle \in \hat{\mathcal{P}}_t\,. \tag{3.2}$$

The machine learning algorithm tunes the model $\mathcal{M}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\phi})$ by adjusting its topology $\boldsymbol{\theta}$ and parameters $\boldsymbol{\phi}$. In error-based learning this process involves the calculation of a set-wise error to which each element-wise error contribute to a different degree,

$$E_{\text{tot}} = \sum_{\boldsymbol{x}_i \in \Psi} E\left(\mathcal{M}(\boldsymbol{x}_i, \boldsymbol{\theta}, \boldsymbol{\phi})\right)\,. \tag{3.3}$$

There are many different forms of the set-wise and element-wise errors, E_{tot} and $E(\cdot)$ respectively, but they can be formulated in a more or less similar fashion as above.

If E_{tot} is to be minimized, then $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ will be set in such way that element-wise contributions are as minimal as possible. As outliers, by their own definition, are rare and infrequent, their error contributions could be left to be relatively large as it is more convenient to focus on those that by being more popular, have a larger contribution to E_{tot}.

Therefore, model $\mathcal{M}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\phi})$ would end up representing more accurately elements more densely grouped than those isolated. However, as we already mentioned, in the model-building case, all elements of Ψ are important, and, perhaps, the isolated ones might be even more important than the clustered ones, as they represent locally optimal zones of the objective set that have not been properly explored.

This situation, although more evident in the multi-objective case, is also present in single-objective EDAs, as they share the model-building methods. It can be argued that this problem has remained unnoticed because most single-objective problems have a uni-modal nature, with only one global optima. However, when faced with a multi-modal problem the same issue arises. When new candidate zones of the decision set are discovered and not yet densely populated, they might end up being disregarded by the model-building algorithm. Figure 3.2 illustrates this situation.

3.2 Diversity Loss in MOEDAs

Another weakness of most MOEDAs,and most EDAs, for that matter, is the loss of population diversity (Shapiro, 2006). In this context diversity loss can be defined as the progressive homogenisation of the population. It could be argued that this drawback has an important impact on the results of MOEDAs, as these algorithms have de dual-goal of yielding solutions close to the Pareto-optimal front and also as diverse as possible.

Diversity loss can be attributed to two main causes:

- biased selection processes, and;
- incorrect model building.

As described in the previous chapter, in Section 2.9, the matting selection in EDAs extracts the best subset of the population to build the model. The continuous selection of the best part of the population could lead to a premature homogenisation of the population and, therefore, to the stagnation of the search process. A similar process could place when carrying out environmental selection, as the repeated substitution of the worst individuals with the offspring could have a similar result. This condition is has been documented and dealt with in MOEAs (Ishibuchi et al., 2010; Wallin and Ryan, 2007) and MOEDAs (Ahn and Ramakrishna, 2007, 2008; Branke et al., 2007; Hong et al., 2007; Wallin and Ryan, 2007).

In the second case, the loss of diversity can be traced back to the above-described outliers issue of model-building algorithms and also to the incorrect estimation or sampling of the model. This fact leads us back to the statement referring that model building has not been correctly acknowledged as a different problem with particular requirements.

The repetitive application of an algorithm that disregards outliers tends to generate more individuals in areas of the search space that are more densely represented. Although there have been some proposals to circumvent this problem, we take the view that the ultimate solution is the use of an adequate algorithm.

Similarly, many machine learning approaches "err in the side of caution" when estimating the model parameters, e. g. the variances of a Gaussian model. This approach, although valid in the original context of application, can prompt the loss of population diversity.

Although model building has not yet been properly identified as the responsible of this problem, there have been a number of works that have tried to "patch" current methods and, therefore make them more suitable for this context. For example, Yuan and Gallagher (2005) proposed a method for avoiding that the variances of a multivariate Gaussian model to drop to "quickly" drop to zero. Similarly, Branke et al. (2007) introduced a permutation sampling that eliminates the sampling errors of UMDA.

Miquélez et al. (2004, 2006a,b) proposed the evolutionary Bayesian classifier-based optimization algorithms. In these algorithms the evolution goes by constructing a Bayesian classifier, but in contrast to EDAs, the selected individuals that are not only the fittest ones. This idea aims at providing faster convergence in optimization problems by modelling the different characteristics that make individuals in the current population fitter or worse using Bayesian classifiers. The algorithms infer classes of individuals with similar finesses and then

samples the Bayesian classifier to produce new individuals that can represent different parts of the fitness spectrum. Thanks to that, these algorithms could circumvent the diversity loss.

There is another strategy for overcoming the loss of diversity because of model construction: to re-inject individuals as the optimization process progresses. This can be carried out in different forms. One example is to resort to hybrid approached like the GA-EDA algorithm (Pena et al., 2004). In GA-EDA variation takes place in both as an EA and an EDA. Therefore it should be more resilient to the diversity loss caused by model building while still having the advantages of an EDA.

These approaches however, do not ultimately address underliying cause of diversity loss, that is, in our opinion, the lack of understanding of the nature of the model-building problem.

3.3 Excessive Computation when Building Models

The third issue to be dealt with is the computational resources wasted on finding an optimal description for the subpopulation being modeled. In the model-building case, optimal model complexity can be sacrificed in the interests of a faster algorithm. This is because the only constraint is to have a model that is sufficiently, but not necessarily optimally, complex to correctly represent the data. This is particularly true when dealing with high-dimensional MOPs, as, in these cases, there will be large amounts of data to be repeatedly processed at each iteration. Even so, most current approaches spend considerable effort on finding optimal model complexity, using minimum description length (Grünwald, 2007), structural risk minimization (Vapnik, 1999), Bayesian information criterion (Schwarz, 1978) or other similar heuristics, as explained in the previous section.

Furthermore, it has been shown that, an excessively accurate or complex model could have a worst performance than simpler ones (Correa and Shapiro, 2006).

In conclusion, we can deduce that an understanding the nature of the model-building problem and the application of suitable algorithms appear to point the way forward in this area.

3.4 A Proof of Concept Experiment

The previously described situation calls for some comparative experiments that correctly expose it. In particular, we will gauge the randomized leader algorithm (Hartigan, 1975), the k-means algorithm (MacQueen, 1967), the expectation maximization algorithm (Dempster et al., 1977), Bayesian networks (Heckerman and Wellman, 1995) and the kernel k-means algorithm (Scholköpf et al., 1998). The first four algorithms were previously used by MOEDAs and the fifth is an advanced clustering algorithm that, in our opinion, could yield interesting results. In order to assess different model-building algorithms, a general EDA framework must be proposed. The model-building algorithms will share this framework. Therefore, it will provide a testing ground common to all approaches and allows us to concentrate only on the topic of interest.

Another measurement dimension that must be quantified is the time complexity of each method under study. In this regard, we determine how many MOEDA iterations were required to reach the solution and how much computation time was consumed on model building in each iteration. This issue has been repeatedly overlooked in MOEDAs studies (and also EDAs', for that matter) since most complexity analysis just report the amount of iterations or the number of evaluations of the objective functions. That implies that the computational cost of model building is left aside when it might be the most complex part of the algorithm as a whole.

3.4.1 General MOEDA Framework

A *general MOEDA framework* must be proposed in order to assess different model-building algorithms. The model-building algorithms will share this framework. Therefore, such a framework will provide a testing ground common to all approaches and we will be able to focus solely on the topic of interest.

Our general MOEDA workflow is similar to other previously existent algorithms, as illustrated in Figure 3.3. It maintains a population of individuals, $\mathcal{P}_t$, where t is the current iteration. It starts with a random initial population $\mathcal{P}_0$ of n_{pop} individuals. It then proceeds to sort the individuals using the NSGA-II fitness assignment function (Deb et al., 2002). This fitness function was chosen because it is in widespread use, although we are aware that better strategies, such as indicator-based options, would probably yield better results.

The fitness function is used to rank individuals according their Pareto dominance relations. Individuals with the same domination rank are then compared using a local crowding distance. This distance favors individuals that are more isolated than those residing in crowded regions of the Pareto front.

A set $\hat{\mathcal{P}}_t$ containing the best $\lceil \alpha \, |\mathcal{P}_t| \rceil$ elements is extracted from the sorted version of $\mathcal{P}_t$,

$$\left|\hat{\mathcal{P}}_t\right| = \lceil \alpha \, |\mathcal{P}_t| \rceil \ . \tag{3.4}$$

Here α is known as the selection percentile.

The model builder under study is then trained using $\hat{\mathcal{P}}_t$ as the training data set. A set of $\lfloor \omega \, |\mathcal{P}_t| \rfloor$ new individuals, which is regulated by the substitution percentile ω, is sampled from the model. Each of these individuals substitutes an individual randomly selected from $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$, which is the section of the population not used for model-building. The output set is then united with the best elements, $\hat{\mathcal{P}}_t$, in order to form the population of the next iteration $\mathcal{P}_{t+1}$.

Iterations are repeated until the given stopping criterion is met. The output of the algorithm is the set of non-dominated solutions from the final iteration, $\mathcal{P}_t^*$.

After some exploratory tests with our EDA, we settled for $\alpha = 0.3$ and $\omega = 0.3$.

3.4.2 Experiment Design

Having properly dealt with the theoretical constituents of our study we will proceed with the study of the performance of each model builder under different circumstances.

1: **Parameters**: n_{pop}, α and ω.
2: $t \leftarrow 0$.
3: Randomly generate initial population, $\mathcal{P}_0$, with n_{pop} individuals.
4: **repeat**
5: Sort $\mathcal{P}_t$ individuals with regard to their fitness function values.
6: Extract the first $\lceil \alpha \, |\mathcal{P}_t| \rceil$ elements of the sorted $\mathcal{P}_t$ to $\hat{\mathcal{P}}_t$.
7: Build model of $\hat{\mathcal{P}}_t$.
8: Sample $\lfloor \omega \, |\mathcal{P}_t| \rfloor$ new individuals from the model.
9: Substitute randomly selected individuals of $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$ with the new individuals to produce $\mathcal{P}'_t$.
10: $\mathcal{P}_{t+1} = \hat{\mathcal{P}}_t \cup \mathcal{P}'_t$.
11: $t \leftarrow t + 1$.
12: **until** end condition met
13: Determine the set of non-dominated individuals of $\mathcal{P}_t$, $\mathcal{P}^*_t$.
14: **return** $\mathcal{P}^*_t$ as the algorithm's solution.

Figure 3.3: Algorithmic representation of the shared MOEDA.

The problems to be addressed are the DTLZ3, DTLZ6 and DTLZ7 scalable continuous test problems (Deb et al., 2004). These problems are discussed in Appendix C. Each problem is solved with an progressive increase of its complexity, as they are configured with 2, 3 and 4 objective functions.

Experiments were carried out under the PISA (Bleuler et al., 2003) framework. Experiments we carried out in a Intel Quad-core CPU computer at 3.4GHz with 4GB of memory running the Linux operating system. Each problem/algorithm combination was repeated 30 times in order to have statistically significant results.

Populations sizes were set to increase with regard to the problem dimension, in particular, $n_{\max}$ was set to 200, 400 and 1100 when solving problems with 2, 3 and 4 objective functions, respectively.

The accuracy and adequacy of the solutions was determined with two commonly accepted indicators of multi-objective optimization performance: the additive epsilon indicator and the Pareto-optimal front coverage indicator, both already introduced in Section 2.2.

The additive epsilon indicator measures how close is the local non-dominated front, $\mathcal{P}^*_t$, to the Pareto-optimal front, $\mathcal{O}^*$. In this particular situation it represents the minimum amount that must be added to $\mathcal{P}^*_t$ in order to dominate $\mathcal{O}^*$.

The resource consumption of multi-objective optimizers is often measured regarding the total running time or counting the amount of objective function evaluations. As we are particularly interested in determining the execution cost of the model builder we took the approach of measuring the amount of CPU operations that were dedicated to that task in each iteration. We found this approach to be more sound and reproducible than just measuring the amount of time. Further details on this are provided in Section B.1.

Every run was left to run for what was estimated as an excessive amount of iterations. This was a viable approach has been used by many previous works. To mark the end of the run we chose the iteration that yielded the lowest mean epsilon indicator value with a

window of 20 iterations.

The outcome of the experiments measured with the epsilon indicator and the coverage indicator are summarized on Figures 3.4 and 3.5. These figures display the results in as box-plots. The results for the lowest dimension are relatively similar across the different algorithms. This can be explained by the fact that in relatively low dimensions these problems are relatively tractable and it is not required that model builder play a fundamental role promoting search.

However, when we analyze the outcome for the $M = 4$ problems some interesting conclusions emerge. First, it is noticeable that statistically sound approaches like Bayesian networks yield poorer results when compared to others. It is most interesting how an statistically unsound algorithm like the leader algorithm produces better results than classical ones like Bayesian networks. The methods based on k-means also performed adequately with a slight advantage in some cases for the kernel version.

This leads to the conclusion that the model-building problem has its own particularities that do not conform to the typical statistical or machine learning scheme. This might open a line of research for creating new model builders that deviate themselves from the current approaches.

This conclusion is reinforced when we analyze the mean amount of iterations used by each algorithm, the mean intra-iteration CPU operations used for model building and the mean total CPU operations used by the algorithms on Figure 3.6. It is very illustrative the case of Bayesian networks that, although they require fewer iterations, its mean CPU operations per iteration is the highest and correspondingly the total amount of CPU operations. Again the case of the leader algorithms is rather interestingly because of its low number of CPU operations per iteration.

It is also noticeable the (presumably) exponential increase on the amount of iterations and the CPU consumption as the problem complexity grows. This means that future algorithms should be aware of this problem and at least try to alleviate this growth.

Although more studies are required to achieve a comprehensive understanding of the nature of the model-building problem these results cast some light on the matter.

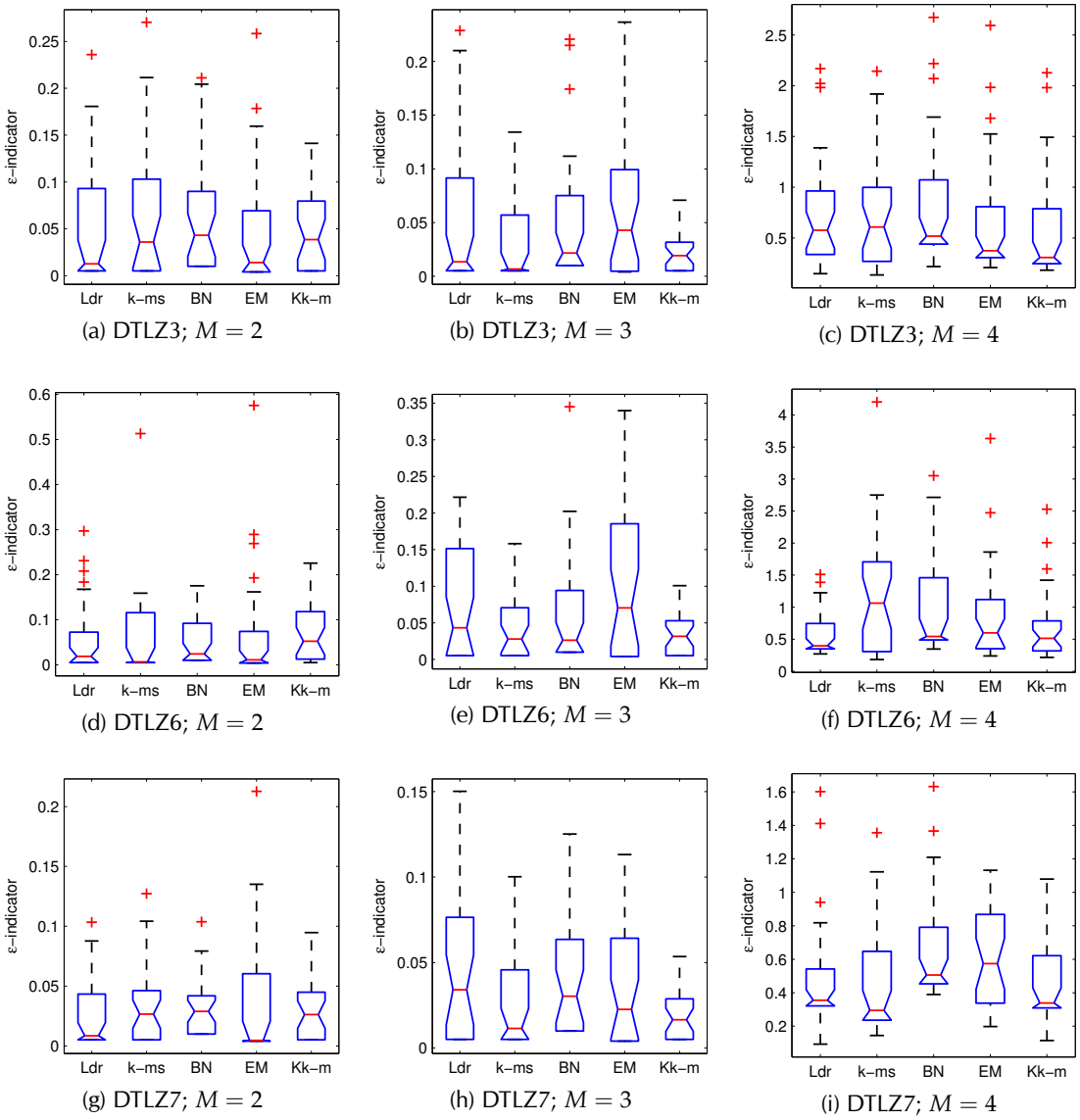
3.5 Remarks on the Experiments

In this study we have taken the first steps towards the understanding of the nature of the model-building problem of MOEDAs. We have found that non-rigorous or inexact approaches performed better than more robust methods. This leads to the conclusion that the model-building issue has its own set of requirements that hinders the application of "classical" methods. Reflecting on this we can see that, for example, outliers must be treated differently as they represent newly discovered local optima that should be explored —not left aside.

However, in order to gain a better comprehension more experiments are necessary. On one hand, different test problems must be addressed to realize if the results obtained here can be generalized. On the other, it is also of interest to further scale the problems to more

objective functions. The analysis of the behavior of the algorithms in those situations might lead to their adaptation to the problem.

The experiences gained here can be used to sketch the requirements for a new model-building algorithm capable of inducing a quantum leap in the performance of MOEDAs and EDAs, for that matter.

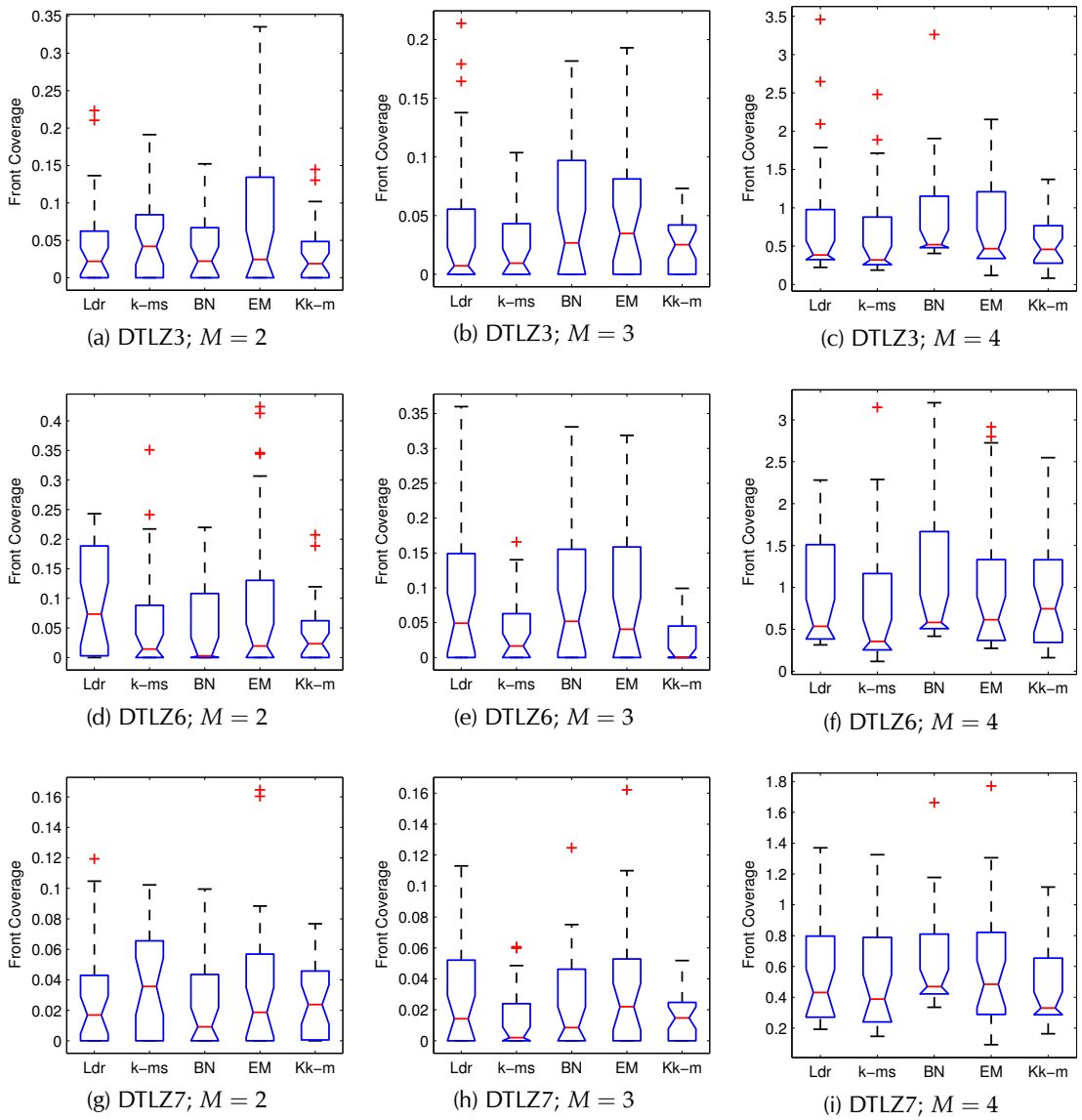


(a) DTLZ3; $M = 2$ (b) DTLZ3; $M = 3$ (c) DTLZ3; $M = 4$

(d) DTLZ6; $M = 2$ (e) DTLZ6; $M = 3$ (f) DTLZ6; $M = 4$

(g) DTLZ7; $M = 2$ (h) DTLZ7; $M = 3$ (i) DTLZ7; $M = 4$

Figure 3.4: Unary additive epsilon indicator values for DTLZ3, DTLZ6 and DTLZ7 problems. Each column show the results obtained with different amount of objectives ($M = 2, 3, 4$) using the Leader algorithm (Ldr), the k–means algorithm (k-ms), Bayesian networks (BN), expectation maximization (EM), and the kernel k–means algorithm (Kk-m).

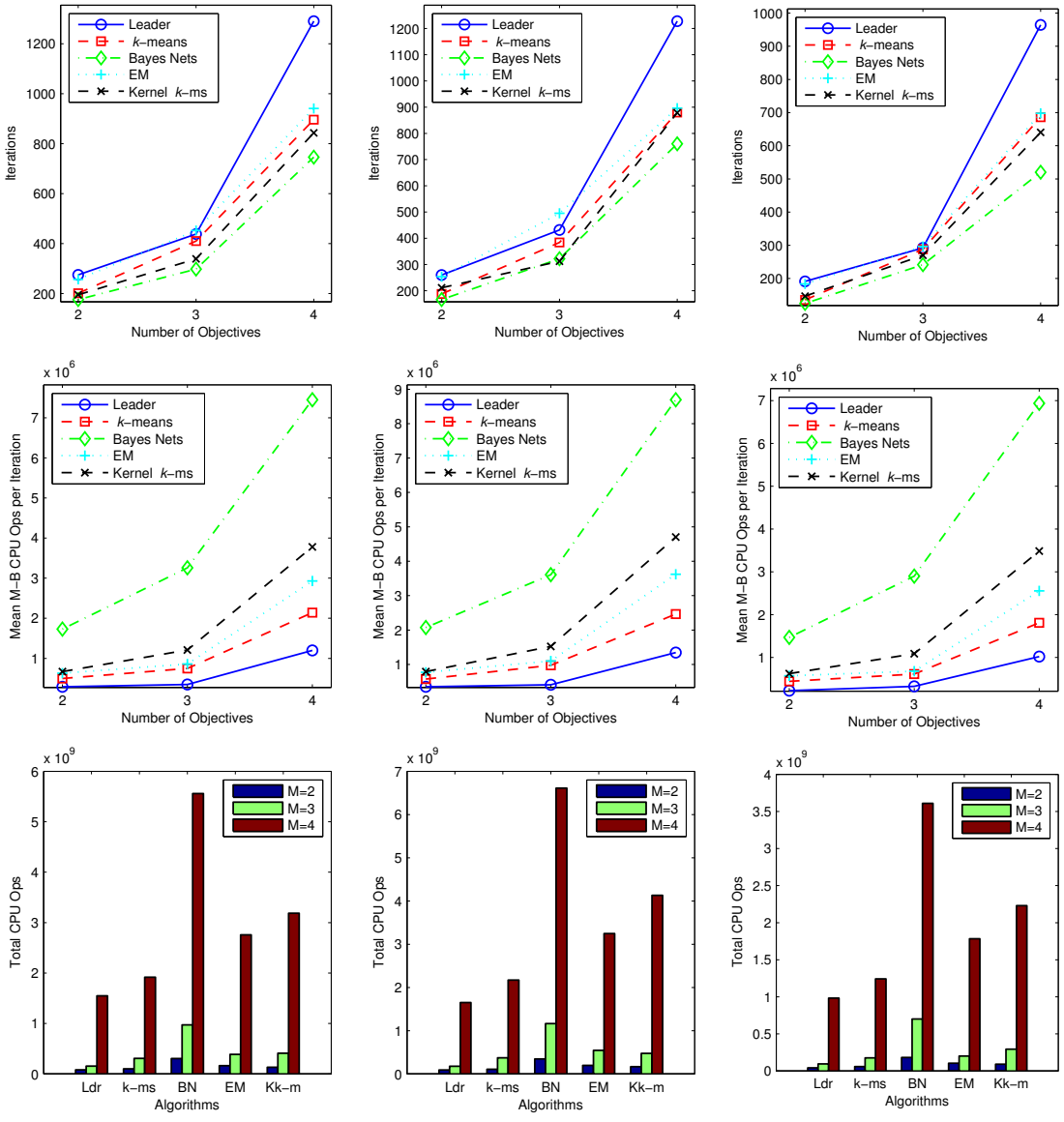


(a) DTLZ3; $M = 2$ (b) DTLZ3; $M = 3$ (c) DTLZ3; $M = 4$

(d) DTLZ6; $M = 2$ (e) DTLZ6; $M = 3$ (f) DTLZ6; $M = 4$

(g) DTLZ7; $M = 2$ (h) DTLZ7; $M = 3$ (i) DTLZ7; $M = 4$

Figure 3.5: Pareto–optimal front coverage for DTLZ3, DTLZ6 and DTLZ7 problems. See Figure 3.4 for details.



(a) DTLZ3 (b) DTLZ6 (c) DTLZ7

Figure 3.6: Comparative analysis of the computational complexity of the algorithms under study. The first row represents the mean amount of iterations used by each algorithm; the second, the mean intra-iteration CPU operations used for model-building; and, the third, the mean total CPU operations used by the algorithms.

4

The Multi-Objective Neural EDA

> I would have preferred to have invented a machine that people could use [...], for example a lawnmower.
>
> *Mikhail Kalashnikov*

IMPROVING the model-building algorithm seems to be a promising direction for research as, to the best of our knowledge; it has not been properly addressed. As a result of a set of preliminary studies presented on the previous chapter where we compared the behavior of a set of model-building algorithms under the same MOEDA framework the need of a fresh approach to model building becomes evident. That study found that, in high-dimensional problems and under the same experimental conditions, statistically robust algorithms, like those commonly used for the synthesis of Bayesian networks, were outperformed by "less robust" approaches like k–means algorithm or the randomized leader algorithm.

The cause of this behavior can be attributed to the fact that statistically rigorous methods are not meant specifically for the problem we are dealing with here. These behaviors, although justified in the original field of application of the algorithms, might hinder the performance of the process, both in the accuracy and in the resource consumption senses. Among these behaviors we can find two important ones: the disregarding of outliers and the dedication of an excessive amount of resources to finding the optimal model structure or topology.

As a conclusion of these analysis and having the previous results as guideline and foundation it becomes obvious that a novel MOEDA, that properly addresses the model-building issue is necessary.

In this chapter we introduce a MOEDA that uses a custom-made model-building algorithm to overcome the problems described here.

4.1 Multi-Objective Neural EDA

The *multi-objective neural estimation of distribution algorithm* (MONEDA) combines the fitness assignment of NSGA-II and a model builder that uses a modification of the growing neural gas network meant for model-building (MB-GNG). The MB-GNG network is a custom-made model-building algorithm devised to cope with the specifications of the task.

The NSGA-II fitness assignment was chosen because of it is very well understood in and its relatively low computational cost. Similarly, using this assignment strategy allows a more direct comparison with similar MOEDA approaches since this is the most popular choice. Nevertheless, it should be pointed out that recent advances in indicator-based fitness assignment, as already mentioned, have rendered that approach an attractive choice. This direction has been started to be explored by the author (Martí et al., 2010b,d).

In brief, MONEDA was devised with the following properties in mind:

- *scalability*: MONEDA is expected to outperform similar algorithms when solving many-objective problems;
- *elitism*: as its has proven itself to be a very advantageous feature in evolutionary algorithms, and;
- *diversity preservation*: in spite of promoting the preservation of the fittest solutions, it is also essential that the population remains as diverse as possible.

4.2 Model Building with a Modified Growing Neural Gas

Clustering algorithms (Xu and Wunsch II, 2008) have been used as part of the model-building algorithms of EDAs and MOEDAs. However, as we discussed in the previous chapter, a custom-made algorithm might be one of the ways of achieving a significant improvement in this field.

The growing neural gas (GNG) network (Fritzke, 1995) has been chosen as a starting point after surveying the literature for suitable candidates. GNG networks are intrinsic self-organizing neural networks based on the neural gas (Martinetz et al., 1993) model. This model relies in a competitive Hebbian learning rule (Martinetz, 1993). The term "neural gas" refers to the behaviour of the center of the nodes during the adaptation process, which distribute themselves like a gas within an imaginary container defined by the bounds implicitly given by the data set on which the network is being trained.

Among the vast number of existing clustering methods we decided to base our approach on GNG because of its interesting properties, in particular:

- the network is sensitive to outliers (Qin and Suganthan, 2004), something undesirable in typical applications but suitable for model-building;
- the network grows to adapt itself automatically to the complexity of the problem being solved;
- it has a fast convergence to low distortion errors and these errors are better than those yielded by "standard" algorithms like k–means clustering, maximum-entropy clustering and Kohonen's self-organizing feature maps (Martinetz et al., 1993);
- its learning rule follows a stochastic gradient descent that follows an explicit energy surface (Qin and Suganthan, 2004);

- although it benefits from the topological ordering of the nodes it does not suffers the problem associated to Kohonen networks, where a node can pull its neighbors to invalid or non-representative locations of the input space, and;
- the addition of a cluster repulsion mechanism fosters the exploration of the input space, making each cluster to represent a distinctive zone of the space.

A GNG network creates an ordered topology of input classes and associates a cumulative error to each. The topology and the cumulative errors are conjointly used to determine how new classes should be inserted. Using these heuristics the model can fit the network dimension to the complexity of the problem being solved. GNG was originally meant for solving unsupervised learning problems (i. e. clustering and vector quantization); it was extended to supervised RBF networks (Flentge, 2006; Fritzke, 1994) to the incremental generation of neuro-fuzzy systems (Fritzke, 1997).

Our model-building GNG (MB-GNG) is an extension of the original GNG. It introduces a cluster repulsion term that fosters a better spread of the clusters along the training data set, as explained by Timm et al. (2004).

MB-GNG is a one-layer network that defines each class as a local Gaussian density and adapts them using a local learning rule. The layer contains a set of nodes $\mathcal{C} = \{c_1, \ldots, c_{N^*}\}$, with $N_0 \leq N^* \leq N_{\max}$. Here N_0 and $N_{\max}$ represent initial and maximal number of nodes in the network.

A node c_i describes a local multivariate Gaussian density that consists of a center, $\boldsymbol{\mu}_i$, and standard deviations vector, $\boldsymbol{\sigma}_i$. It also has an *accumulated error*, ξ_i, and a set of edges that define the set of topological neighbors of c_i, $\mathcal{V}_i$. Each edge has an associated age, ν_{ij}.

MB-GNG creates a quantization of the inputs space using a modified version of the GNG algorithm and then computes the deviations associated to each node.

The dynamics of a GNG network consists of three concurrent processes: network adaptation, node insertion and node deletion. The combined use of these three processes renders GNG training Hebbian in spirit (Martinetz, 1993).

The network is initialized with N_0 nodes with their centers set to randomly chosen inputs. A training iteration starts after an input $\boldsymbol{x}$ is randomly selected from the training data set. Then two nodes are selected for being the closest ones to $\boldsymbol{x}$. The *best-matching node*, c_b,

$$b = \arg\min_{i=1,..,N^*} d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right), \tag{4.1}$$

is the closest node to $\boldsymbol{x}$. Consequently, the *second best-matching node*, $c_{b'}$, is determined as

$$b' = \arg\min_{i=1,..,N^*; i \neq b} d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right). \tag{4.2}$$

Here $d\left(\boldsymbol{a}, \boldsymbol{b}\right)$ is a distance metric. For this work we have used $d(\cdot)$ defined as

$$d\left(\boldsymbol{a}, \boldsymbol{b}\right) = \left\|\boldsymbol{a} - \boldsymbol{b}\right\|. \tag{4.3}$$

If $c_{b'}$ is not a neighbor of c_b then a new edge is established between them $\mathcal{V}_b = \mathcal{V}_b \cup \{c_{b'}\}$ with zero age, $\nu_{bb'} = 0$. If, on the other hand, $c_{b'} \in \mathcal{V}_b$ the age of the corresponding edge is reset $\nu_{bb'} = 0$.

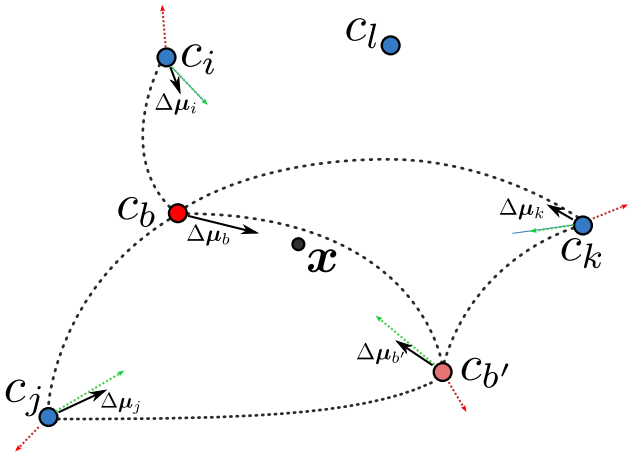


Figure 4.1: Schematic representation of MB-GNG learning. Node neighborhood edges are represented by the dotted arcs. The center of the best-matching node, $\boldsymbol{\mu}_b$, is modified according to (4.5). For the neighbors of c_b, learning takes place according to 4.6. For those nodes the learning rule combines a movement towards the input (represented in green) and a repulsion term which takes into account the distance to neighbor nodes (in red). This repulsion term avoids the concentration of nodes in the same zone of the data space. No learning takes place on nodes disconnected from c_b, like, in this case, c_l

At this point, the age of all edges is incremented by one. If an edge is older than the maximum age, $v_{ij} > v_{\max}$, then the edge is removed. If a node becomes isolated from the rest, it is also deleted.

Clustering error is then added to the best-matching node error accumulator,

$$\Delta\xi_b = d\left(\boldsymbol{\mu}_i, \boldsymbol{x}\right)^2 . \tag{4.4}$$

After that, learning takes place in the best-matching node and its neighbors with rates ϵ_{best} and ϵ_{vic} ($\epsilon_{\text{best}} > \epsilon_{\text{vic}}$), respectively. These two rates gate the movement of the centers of the nodes involved towards the current input $\boldsymbol{x}$. This process is presented in an schematic form on Figure 4.1.

For c_b, adaptation follows the rule originally used by GNG,

$$\Delta\boldsymbol{\mu}_b = \epsilon_{\text{best}}\left(\boldsymbol{x} - \boldsymbol{\mu}_b\right) . \tag{4.5}$$

However, for the neighbors of c_b, a cluster repulsion term (Timm et al., 2004) is added to the original formulation. For those nodes the learning rule combines a movement towards the input and a repulsion term which takes into account the distance to neighbor nodes. This repulsion term avoids the meaningless concentration of nodes in the data space and therefore, promotes a proper representation of the data set with fewer nodes.

Following that, the learning rule for those nodes can be expressed as, $\forall c_v \in \mathcal{V}_b$,

$$\Delta\boldsymbol{\mu}_v = \epsilon_{\text{vic}}\left(\boldsymbol{x} - \boldsymbol{\mu}_v\right) + \beta e^{\left(-\frac{d(\boldsymbol{\mu}_v,\boldsymbol{\mu}_b)}{\zeta}\right)} \frac{\sum_{c_u \in \mathcal{V}_b} d\left(\boldsymbol{\mu}_u, \boldsymbol{\mu}_b\right)}{|\mathcal{V}_b|} \frac{\left(\boldsymbol{\mu}_v - \boldsymbol{\mu}_b\right)}{d\left(\boldsymbol{\mu}_v, \boldsymbol{\mu}_b\right)} . \tag{4.6}$$

This approach was already used as part of the robust GNG (Qin and Suganthan, 2004) and it has proven itself useful for obtaining a good spread of the clusters in the inputs' space.

In the aforementioned work, it was stated that the adaptation rule is not sensitive to its parameters. Here β is an integral multiplier that defines the amplitude of the repulsive force while ζ controls the weakening rate of the repulsive force regarding the distance between the nodes' centers. We have set them to $\beta = 2$ and $\zeta = 0.1$ as suggested by Qin and Suganthan (2004).

After a given number, T_+, of data set iterations (epochs, in the neural networks terminology) have taken place, it can be presumed that there is enough information stored in the error accumulators, ξ_i. This information is used to determine where to add new nodes to the network. In particular, if the current iteration is an integer multiple of T_+ and the network has not reached its maximum size ($N^* < N_{\max}$) then a new node is inserted to the network.

First, the node with the largest error, c_e, is selected. Then, the worst node among its neighbors, $c_{e'}$, is located. Then N^* is incremented and the new node, c_{N^*}, is inserted between the two nodes,

$$\boldsymbol{\mu}_{N^*} = 0.5\,(\boldsymbol{\mu}_e + \boldsymbol{\mu}_{e'})\,. \tag{4.7}$$

The edge between c_e and $c_{e'}$ is removed and two new edges connecting c_{N^*} with c_e and $c_{e'}$ are created. The accumulated errors are decreased

$$\xi_e = \delta_I \xi_e,\ \xi_{e'} = \delta_{\mathrm{I}} \xi_{e'}, \tag{4.8}$$

by a rate $0 \leq \delta_{\mathrm{I}} \leq 1$. The error of the newly created node is computed as

$$\xi_{N^*} = 0.5(\xi_e + \xi_{e'}). \tag{4.9}$$

Finally, the errors of all nodes are decreased by a factor δ_{G},

$$\xi_i = \delta_{\mathrm{G}} \xi_i,\ i = 1, .., N^*. \tag{4.10}$$

Stopping the learning of GNG is a non-trivial issue shared by the rest of clustering algorithms and all reiterative heuristic algorithms. As the main priority here is covering the input space as much as possible we will stop if, after a learning epoch, the standard deviation of the accumulated errors is smaller than a certain threshold, ρ,

$$\sqrt{\frac{1}{N^*} \sum_{i=1}^{N^*} (\xi_i - \overline{\xi})^2} < \rho. \tag{4.11}$$

This means that it will stop when the outliers are as well represented as possible.

After training has ended the deviations, σ_i, of the nodes must be computed. For this task we employ the unbiased normal estimator of the deviations (Schervish, 1997) detailed in the following algorithm:

Set $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{N^*} = \mathbf{0}$ and $n_1, \ldots, n_{N^*} = 0$.
for all $\boldsymbol{x} \in \Psi$ **do**
  Determine the closest node, c_c to $\boldsymbol{x}$.
  $\boldsymbol{s}_c = \boldsymbol{s}_c + (\boldsymbol{x} - \boldsymbol{\mu}_c)^2$.
  $n_c = n_c + 1$.
end for
Compute the deviations as $\boldsymbol{\delta}_i = \sqrt{\frac{s_i}{n_i}}$.

The local Gaussian densities resulting from the described algorithm can be combined to synthesize the Gaussian mixture with parameters $\boldsymbol{\Theta}$,

$$P(\boldsymbol{x}|\boldsymbol{\Theta}) = \frac{1}{N^*} \sum_{i=1}^{N^*} P(\boldsymbol{x}|\boldsymbol{\mu}_i, \sigma_i) . \tag{4.12}$$

Each Gaussian density is formulated as

$$P(\boldsymbol{x}|\boldsymbol{\mu}_i, \sigma_i) = \frac{1}{(2\pi)^{n/2}|\boldsymbol{\Sigma}_i|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_i})^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right) , \tag{4.13}$$

with the covariance matrices $\boldsymbol{\Sigma}_i$ defined as a diagonal matrix with its non-zero elements set to the values of the deviations σ_i,

$$\boldsymbol{\Sigma}_i = \boldsymbol{I}\sigma_i . \tag{4.14}$$

Here $|\Sigma_i|$ is the determinant of $\boldsymbol{\Sigma}_i$, $\boldsymbol{I}$ is the identity matrix and, again, n is the dimension of $\boldsymbol{x}$.

The Gaussian mixture can be used by the EDA to generate new individuals. This new individuals are created by sampling the $P(\boldsymbol{x}|\boldsymbol{\Theta})$. The generation of randomly distributed numbers that follow a given distribution has been dealt in depth by many authors. It has been properly described, for example, by Rubinstein (1981). In our case, we applied the Box-Muller transformation (Box and Muller, 1958). This transformation converts uniformly distributed random variables to a new set of random variables with a Gaussian distribution.

See Figure 4.2 for a summary of the MB-GNG algorithm.

4.3 The MONEDA Algorithm

MONEDA maintains a population, $\mathcal{P}_t$, of n_{pop} individuals; where t is a given iteration. The algorithm's workflow is similar to other EDAs (see Figure 4.3). It starts with a random initial population $\mathcal{P}_0$ of individuals. It then proceeds to sort the individuals using the NSGA-II fitness assignment function.

The NSGA-II fitness assignment takes place in two phases. First, individuals are ranked according the dominance relations established between them. After that, Individuals with the same domination rank are then compared using a local crowding distance.

The first step consists in classifying the individuals in a series of categories, $\mathcal{F}_1, \ldots, \mathcal{F}_L$. Each of these categories stores individuals that are only dominated by the elements of the previous categories,

$$\begin{aligned} \forall \boldsymbol{x} \in \mathcal{F}_i : \quad & \exists \boldsymbol{y} \in \mathcal{F}_{i-1} \text{ such that } \boldsymbol{y} \prec \boldsymbol{x}, \text{ and;} \\ & \nexists \boldsymbol{z} \in \mathcal{P}_t \setminus (\mathcal{F}_1 \cup \ldots \cup \mathcal{F}_{i-1}) \text{ that } \boldsymbol{z} \prec \boldsymbol{x}; \end{aligned} \tag{4.15}$$

with $\mathcal{F}_1$ equal to $\mathcal{P}_t^*$, the set of non-dominated individuals of $\mathcal{P}_t$.

After all individuals are ranked, a local crowding distance is assigned to them. The use of this distance primes individuals more isolated with respect to others. The assignment process goes as follows,

for all category sets $\mathcal{F}_l$, having $f_l = |\mathcal{F}_l|$ **do**

for all individuals $\boldsymbol{x}_i \in \mathcal{F}_l$ **do**
$d_i = 0.$
end for
for all objective functions $m = 1, \ldots, M$ **do**
$\boldsymbol{I} = \text{sort}\,(\mathcal{F}_l, m)$ (generate index vector in ascending order).
$d_{I_1}^{(l)} = d_{I_{f_l}}^{(l)} = \infty.$
for $i = 2, \ldots, f_l - 1$ **do**
Update the remaining distances as

$$d_i = d_i + \frac{f_m\left(\boldsymbol{x}_{I_{i+1}}\right) - f_m\left(\boldsymbol{x}_{I_{i-1}}\right)}{f_m\left(\boldsymbol{x}_{I_{f_l}}\right) - f_m\left(\boldsymbol{x}_{I_1}\right)}\,.$$

end for
end for
end for

Here the sort $(\mathcal{F}, m)$ function produces an ascending ordered index vector $\boldsymbol{I}$ with respect to the value of objective function f_m.

Having the individual ranks and their local distances, they are sorted using the following operator:

Definition 4.1 (crowded comparison operator) *An individual $\boldsymbol{x}_i$ is better than $\boldsymbol{x}_j$ if:*

- *$\boldsymbol{x}_i$ has a better rank: $\boldsymbol{x}_i \in \mathcal{F}_k$, $\boldsymbol{x}_j \in \mathcal{F}_l$ and $k < l$, or;*
- *if $k = l$ and $d_i > d_j$.*

A set $\hat{\mathcal{P}}_t$ containing the best $\lfloor \alpha\,|\mathcal{P}_t| \rfloor$ elements is extracted from the sorted version of $\mathcal{P}_t$,

$$\left|\hat{\mathcal{P}}_t\right| = \lfloor \alpha\,|\mathcal{P}_t| \rfloor = \left\lfloor \alpha n_{\text{pop}} \right\rfloor\,. \tag{4.16}$$

A MB-GNG network is then trained using $\hat{\mathcal{P}}_t$ as its training data set. In order to have a controlled relation between size of $\hat{\mathcal{P}}_t$ and the maximum size of the network, $N_{\max}$, these two sizes are bound by the rate $\gamma \in (0, 1]$,

$$N_{\max} = \left\lceil \gamma \left|\hat{\mathcal{P}}_t\right| \right\rceil = \left\lceil \gamma \left\lfloor \alpha n_{\text{pop}} \right\rfloor \right\rceil\,. \tag{4.17}$$

The trained GNG network is a model of $\hat{\mathcal{P}}_t$. The network can be interpreted as a Gaussian mixture, as explained in the previous section. Therefore it can be used to sample new individuals. In particular, $\lfloor \omega\,|\mathcal{P}_t| \rfloor$ new individuals are synthesized.

Each one of these individuals substitutes a randomly selected ones from the section of the population not used for model-building $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$. The set obtained is then united with the best elements, $\hat{\mathcal{P}}_t$, to form the population of the next iteration $\mathcal{P}_{t+1}$. Some other substitution strategies could be used in this step. For example, the new individuals could substitute the worst individuals of $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$. We have chosen the previously described approach because it promotes diversity and avoids stagnation.

Iterations are repeated until a given stopping criterion is met. The output of the algorithm is a subset of $\mathcal{P}_t$ that contains the non-dominated solutions, $\mathcal{P}_t^*$.

The outline of MONEDA is presented in algorithmic form on Figure 4.4.

Parameters:
- N_0 and $N_{\max}$, bounds on the number of nodes.
- $\nu_{\max}$, maximum edge age.
- ϵ_{best} and ϵ_{vic}, learning rates.
- T_+, number of dataset iterations before node insertion process.
- δ_{I}, δ_{G}, error redistribution rates.
- ρ, stopping threshold.

Initialize $t \leftarrow 0$.
Randomly select N_0 elements from the dataset and initialize the same number of nodes using those elements as centers $\boldsymbol{\mu}$, and empty $\mathcal{V}$s.
repeat
  Determine best-matching node, c_b, following (4.1).
  Determine second best-matching node, $c_{b'}$, following (4.2).
  if $c_{b'} \notin \mathcal{V}_b$ **then**
    Make c_b and $c_{b'}$ neighbors,

$$\mathcal{V}_b = \mathcal{V}_b \cup \{c_{b'}\}; \mathcal{V}_{b'} = \mathcal{V}_{b'} \cup \{c_b\}.$$

  end if
  Neighborhood edge age is set to $\nu_{bb'} = 0$.
  Update c_b error accumulator, ξ_b, according to (4.4).
  Learning takes place in c_b as specified in (4.5).
  $\forall c_{\text{vic}} \in \mathcal{V}_b$ learning is carried out according to (4.6).
  if $t \bmod T_+ = 0$ **and** $N^* < N_{\max}$ **then**
    Determine node with the largest accumulated error, c_e, and the worst among its neighbors, $c_{e'}$, $c_{e'} \in \mathcal{V}_e$.
    Dissolve edge between c_e and $c_{e'}$,

$$\mathcal{V}_e = \mathcal{V}_e \setminus \{c_{e'}\}; \mathcal{V}_{e'} = \mathcal{V}_{e'} \setminus \{c_e\}.$$

    Create a new node between c_e and $c_{e'}$, as in (4.7) and (4.9).
    Decrease c_e and $c_{e'}$ accumulated errors, as expressed in (4.8).
    Decrease the errors of the remaining nodes, following (4.10).
  end if
until inequality (4.11) holds.
Compute the unbiased estimator of the deviations.

Figure 4.2: Model-building growing neural gas (MB-GNG) algorithm.

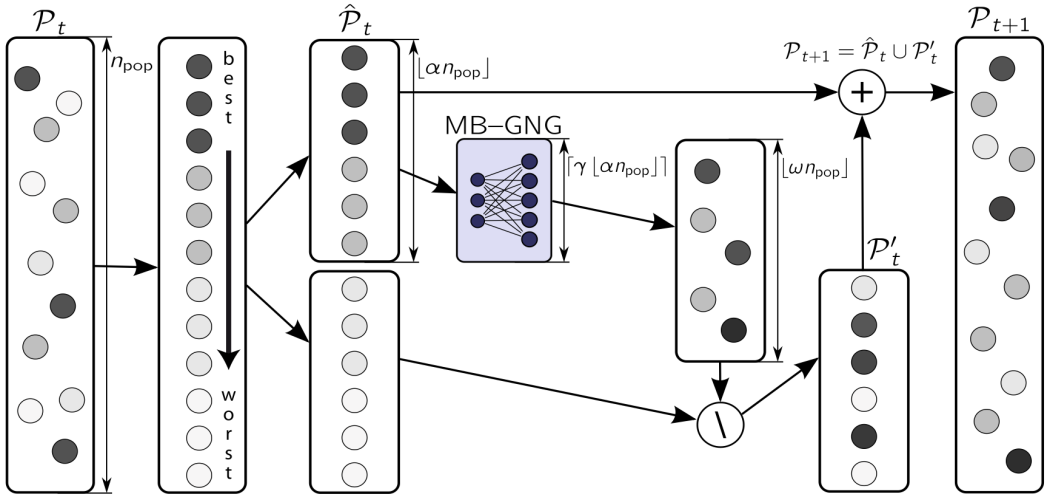


Figure 4.3: Diagram representation of the MONEDA algorithm. At iteration t the population is ranked using the fitness assignment function. Then a population subset $\hat{\mathcal{P}}_t$ containing the best $\lfloor \alpha n_{\text{pop}} \rfloor$ elements of $\mathcal{P}_t$ is extracted. A MB-GNG network is trained with the elements of $\hat{\mathcal{P}}_t$. $\lfloor \omega n_{\text{pop}} \rfloor$ new individuals are sampled from the neural network. These individuals substitute the same amount of randomly selected elements of $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$. The resulting set is then combined with $\hat{\mathcal{P}}_t$ to form the population of the next iteration, $\mathcal{P}_{t+1}$.

1: **MB-GNG parameters**: N_0, $\nu_{\max}$, ϵ_{best}, ϵ_{vic}, T_+ δ_{I}, δ_{G} and ρ.
2: **MONEDA parameters**: n_{pop}, α, γ and ω.
3: $t \leftarrow 0$.
4: Randomly generate the initial population $\mathcal{P}_0$ with n_{pop} individuals.
5: **repeat**
6: Sort $\mathcal{P}_t$ individuals with regard to the crowded comparison operator.
7: Extract first $\alpha \left|\mathcal{P}_t\right|$ elements the sorted $\mathcal{P}_t$ to $\hat{\mathcal{P}}_t$.
8: Train MB-GNG network with $\hat{\mathcal{P}}_t$ training data set and $N_{\max} = \left\lfloor \gamma \left|\hat{\mathcal{P}}_t\right| \right\rfloor$ (see algorithm in Figure 4.2).
9: Sample $\lfloor \omega \left|\mathcal{P}_t\right| \rfloor$ from the MB-GNG.
10: Substitute randomly selected individuals of $\mathcal{P}_t \setminus \hat{\mathcal{P}}_t$ with the new individuals to produce $\mathcal{P}'_t$.
11: $P_{t+1} = \hat{\mathcal{P}}_t \cup \mathcal{P}'_t$.
12: $t = t + 1$.
13: **until** end condition not met
14: Determine the set of non-dominated individuals of $\mathcal{P}_t$, $\mathcal{P}^*_t$.
15: **return** $\mathcal{P}^*_t$ as the algorithm's solution.

Figure 4.4: Algorithmic representation of MONEDA.

5

The MGBM Stopping Criteria for Multi-Objective Optimization

> There's no sense in being precise when you don't even know what you're talking about.
>
> *John von Neumann*

As already stated in the introduction, the stopping criterion issue has been repeatedly named as one of the key topics requiring proper attention in the multi-objective optimization and MOEAs research areas (Coello Coello, 2000, 2004). Even so, it has been continually neglected. This is not surprising since this matter plays a secondary role compared with the main lines of research in the area. In face of complex real-world problems, though, the lack of a firm theoretical understanding of the problem stands in the way of finding appropriate solutions.

In this chapter we put forward a comprehensive study of the design of a global stopping criteria for multi-objective optimization. We propose a global stopping criterion, which we have called MGBM after the authors last names. MGBM combines a novel progress indicator, named mutual domination rate (MDR) indicator, with a simplified Kalman filter (Kalman, 1960), which is used as an evidence-gathering process. The MDR indicator, which is also introduced here, is a special-purpose solution designed to deal with stopping. It is capable of gauging the progress of the optimization at a low computational cost and is therefore suitable for solving complex or many-objective problems (Purshouse, 2003).

The main contributions of this chapter can be summarized as:

- detailed discussion of the stopping criterion issue and its current state, requirements and problem-solving strategies;
- discussion of different approaches for addressing this issue, and;
- the proposal and testing of a novel stopping criterion.

It should be noted that, although the criteria discussed here are meant for MOPs and MOEAs, they could be easily adapted to other soft computing or numerical methods by replacing the local improvement metric as appropriate.

5.1 The MGBM Stopping Criterion

As mentioned in Section 2.11, a stopping criterion should be composed of two components. One component measures the improvement in the solutions obtained after an iteration and the other keeps track of these measurements in order to decide whether or not the execution of the algorithm should be stopped.

MGBM is a global criterion that combines a local improvement indicator, called the mutual domination rate (MDR) indicator, and a global evidence-gathering criterion that decides when the evolution of values yielded by the local metric indicates that the algorithm should be stopped. The local indicator contrasts the non-dominated individuals of the current and preceding iterations in order to compute a measure of the improvement produced by the current iteration. This indicator is discussed in detail in the next section. The evidence-gathering process tracks the values of the indicator across iterations using a Kalman filter (described in section 5.3). The Kalman filter settings are unusual, as it is not designed to predict the outcome of the indicator across iterations. Instead, it is used to detect situations where no further progress will be made. Section 5.3.2 presents the fine points of this matter.

5.2 Mutual Domination Rate Indicator

Intuitively, the performance assessment area, briefly introduced in Section 2.2 a natural grounding for addressing this issue. Performance indicators, although designed to determine how similar a solution is to the Pareto-optimal front, can be reformulated to compare two solutions output by two consecutive iterations. The main drawback of directly applying this class of solution is the high computational complexity of the indicators.

The solution to this problem is to create an indicator from scratch designed specially for the intended purpose. This was the idea that prompted the formulation of the *mutual domination rate indicator* (MDR).

To measure the progress of the evidence-gathering process, we use a metric based on the set of non-dominated solutions of two consecutive iterations, $\mathcal{P}_t^*$ and $\mathcal{P}_{t-1}^*$.

In order to simplify the explanation we introduce the $\Delta(\mathcal{A}, \mathcal{B})$ function that returns the set of elements of $\mathcal{A}$ that are dominated by at least one element of $\mathcal{B}$. Expressed more formally,

$$\mathcal{C} = \Delta(\mathcal{A}, \mathcal{B}) \,, \tag{5.1}$$

such that

$$\forall \boldsymbol{x} \in \mathcal{C}, \boldsymbol{x} \in \mathcal{A}, \text{ and } \exists \boldsymbol{y} \in \mathcal{B} \text{ with } \boldsymbol{y} \prec \boldsymbol{x} \,. \tag{5.2}$$

The progress indicator $I_{\text{mdr}}(t) \in [-1, 1]$ contrasts how many non-dominated individuals of iteration t dominate the non-dominated individuals of the previous iteration $(t-1)$ and vice versa,

$$I_{\text{mdr}}(\mathcal{P}_t^*, \mathcal{P}_{t-1}^*) = \frac{\|\Delta(\mathcal{P}_{t-1}^*, \mathcal{P}_t^*)\|}{\|\mathcal{P}_{t-1}^*\|} - \frac{\|\Delta(\mathcal{P}_t^*, \mathcal{P}_{t-1}^*)\|}{\|\mathcal{P}_t^*\|} \,. \tag{5.3}$$

The I_{mdr} indicator provides different types of information. If $I_{\text{mdr}} = 1$, the entire population of iteration t is better than its predecessor. If $I_{\text{mdr}} = 0$, there has not been any substantial

progress. The worst case, $I_{\text{mdr}} = -1$, indicates that iteration t has not improved any of the solutions of its predecessor.

As we mentioned earlier, a stopping criterion should be able to discover three situations where the algorithm execution should be stopped. From the stopping point of view, all these situations can be interpreted as whether or not the algorithm has made progress. Note that if we focus on the dominance relation, we disregard the spread of elements along the Pareto-optimal front. This means that the algorithm will be able to detect when there is no improvement towards this front but not when the spread of solutions along the Pareto-optimal front is better.

The determination of the non-dominated individuals after each iteration can be computationally expensive. As most MOEAs extract such individuals for their own purposes, however, it would be reasonable to embed this part of the criterion into the actual MOEAs.

Having the current and previous non-dominated sets $\mathcal{P}_t^*$ and $\mathcal{P}_{t-1}^*$, the order of complexity of calculating I_{mdr} is $O\left(M \cdot |\mathcal{P}_t^*| \cdot \left|\mathcal{P}_{t-1}^*\right|\right)$.

5.3 Gathering Evidence

Our approach is based on the recursive estimation prediction and update framework proposed by Kalman filters. For this reason, we will assume that there is no correlation between the noise present in the measured progress indicator in consecutive iterations. Furthermore, the estimated value of the progress indicator and its associated covariance are governed by a Markov process, and therefore the outcome of each iteration depends on the previous iteration only.

5.3.1 Kalman Filters

The *Kalman filter* (Kalman, 1960; Maybeck, 1979) provides an efficient computational means to estimate the state of a dynamic system from a series of incomplete and noisy measurements. This filter is the linear estimator with minimum squared error that can be applied to any dynamic system with errors following any distribution where the two first moments of the distribution are known. Furthermore, if we know that probability distributions are Gaussian and the system dynamics are linear, the Kalman filter is the globally optimal state estimator. It is very powerful since it supports estimations of past, current, and future states, even when some aspects of the modelled system are unknown.

The Kalman filter addresses the general problem of estimating the state of a discrete-time controlled process that is ruled by a linear stochastic difference equation.

The state of the filter is represented by two variables:

- $\hat{x}_t$, the estimate of the state at time t, and
- P_t, the error covariance matrix, which is a measure of the estimated accuracy of the current state estimate.

The Kalman filter estimates a process state by a recursive feedback control that can be separated in the *prediction* and *update* phases.

The prediction phase is responsible for making an *a priori estimation* of the future state of the system relying on the current state and error covariance estimates. The update phase is responsible for feeding back the (noisy) measurement of the state of the system to output an improved *a posteriori estimate*.

The Kalman filter assumes a dynamic model given by

$$x_t = Ax_{t-1} + Bu_t + w_t\,, \tag{5.4}$$

where u_t is an optional control input and the random variables $w_t \sim N(0, Q)$ represent the process noise.

Additionally, the measurement process is modeled by

$$z_t = Hx_t + v_t\,, \tag{5.5}$$

where H relates the real state of the process x_t to the measurement z_t and $v_t \sim N(0, R)$ is the measurement noise.

First, the a priori estimation, $\hat{x}_t^-$, and its error covariance, P_t^-, are calculated as

$$\hat{x}_t^- = A\hat{x}_{t-1} + Bu_t\,, \tag{5.6}$$
$$P_t^- = AP_{t-1}A^T + Q\,. \tag{5.7}$$

Then the update phase proceeds by computing the Kalman gain,

$$K_t = \frac{P_t^- H^T}{HP_t^- H^T + R}\,. \tag{5.8}$$

The a posteriori estimation is calculated as the feedback is entered in the filter as

$$\hat{x}_t = \hat{x}_t^- + K_t\left(z_t - H\hat{x}_t^-\right)\,. \tag{5.9}$$

Finally, an a posteriori error covariance estimate is output by

$$P_t = (I - K_t H)P_t^-\,, \tag{5.10}$$

where I is the identity matrix.

5.3.2 Using Kalman Filters to Gather Evidence

The application of Kalman filters is an unconventional approach to evidence gathering. Instead of trying to predict the outcome of a given variable across time, we are interested in detecting when a variable (in this case the MDR indicator) has stabilized around zero. Due to its recursive formulation, the estimated variable at time t summarizes all the evidence gathered until then, plus the associated covariance error, P_t, which would be the minimum possible error under linear conditions.

For this reason, in our case, we keep track of the algorithm progress indicator value at iteration t, $I_{\text{mbr}}(\mathcal{P}^*_t, \mathcal{P}^*_{t-1})$.

After each iteration, we compute the *a priori* estimated indicator $\hat{I}^-_t$ using a simple version of the dynamic model (5.4) with $A = 1$ and $B = 0$. This implies that we are taking a positivist stance and predicting that the indicator will remain constant across iterations and, therefore, be equal to the *a posteriori* estimation, $\hat{I}_{t-1}$,

$$\hat{I}^-_t = \hat{I}_{t-1}\,, \tag{5.11}$$

disregarding the control input, u, as there is no direct information on changes of I.

As this is a purely computational process, we can disregard the prediction error in our dynamic model ($Q = 0$). The neglect of the plant noise covariance matrix Q is assumable, provided that the deviation from the linear approximation is much less than the observation process (R). This is valid if the dynamics of the convergence process is smooth, which will be true in the last phase of convergence of any search algorithm under normal conditions. On the other hand, under these assumptions, we would not be rigorously modeling the behavior during the initial transient period. However, as the estimation is not intended to be used to track the values of I, but to detect when it has reached a stable state that represents algorithm stagnation, we are interested in having a precise model for the latter part of the convergence process.

Correspondingly, the a priori error covariance becomes

$$P^-_t = P_{t-1}\,. \tag{5.12}$$

We then rewrite (5.5) as

$$\begin{aligned} z_t &= I_{\text{mdr}}\,(\mathcal{P}^*_t, \mathcal{P}^*_{t-1}) && (5.13)\\ &= I_t + v_t\,, v_t \sim N(0, R), && (5.14) \end{aligned}$$

where $I_{\text{mdr}}(\mathcal{P}^*_t, \mathcal{P}^*_{t-1})$ is calculated following (5.3). Here we assume that $I_{\text{mdr}}()$ is affected by a Gaussian process attributable to the search process taking place as part of the evolutionary algorithm.

The correction step of the process becomes

$$K_t = \frac{P^-_t}{P^-_t + R}\,. \tag{5.15}$$

Here R can be interpreted as the rate at which the criterion will take into account a single measurement and therefore provide a faster reaction to changes or if, on the contrary, the criterion is biased toward a more global (or more inertial) approach.

Therefore, the a posteriori estimation of the indicator can be expressed as the current result of the indicator

$$\hat{I}_t = \hat{I}^-_t + K_t(z_t - \hat{I}^-_t)\,. \tag{5.16}$$

The above assumptions merit further discussion, as they imply an alternative use of Kalman filters and are, therefore, likely to lead to a misunderstanding of the inner workings of the criterion. As already discussed, Kalman filters are generally used for estimating

the time-sequence values of a definite variable from a set of (noisy) measurements. This is not the case here. In this case, we have configured the filter to capture a "no-progress state" where the evolutionary search process has stagnated. Therefore, the criterion recognizes when the dynamics of the evolutionary process matches the no-progress state represented by the filter.

One main concern is to assure that the algorithm will not stop too early. It can be assumed that there will be temporary stagnation scenarios in the early stages of the evolutionary process that should not be taken into account. The value of R plays a key role for this purpose. By assuming $Q = 0$, P_t^- and K_t will converge to zero at a rate that depends only on R. This implies that R controls how sensitive the criterion will be in the initial part of the execution. This assumption has been validated in (Guerrero et al., 2009), where we analyzed different alternatives for Q and some adaptive configurations. A noteworthy conclusion drawn from the above research is that the selection of different values of Q had little impact on the stopping decision.

Figure 5.1 contains plots of the values of the MDR indicator, $I_{\text{mdr}}\left(\mathcal{P}_t^*, \mathcal{P}_{t-1}^*\right)$, and the a priori and a posteriori estimations, $\hat{I}_t^-$ and $\hat{I}_t$, used in an NSGA-II run to solve the DTLZ3 problem (see Chapter 6 for details). These plots illustrate how the three values interact with each other across the algorithm iterations and how the a posteriori indicator smooths out the readings yielded by the indicator with a definite shift in time.

There is more than one situation where the values of the indicator and the estimators suggest that algorithm execution can be safely stopped. For the MGBM criterion we chose a scheme that will activate if the a posteriori estimation $\hat{I}_t$ and associated confidence interval falls below a definite threshold

$$\hat{I}_t + 2\sqrt{P_t} < \hat{I}_{\min}\,. \tag{5.17}$$

In particular, as we are interested in stopping when no further progress is predicted, it should stop in a situation that is represented by $\hat{I}_{\min} = \varepsilon$, with $\varepsilon \to 0$.

5.4 Algorithmics of the Criterion

Relying on the equations introduced above we can formulate the algorithmic scheme of the *MGBM stopping criterion*. This algorithm is outlined in Figure 5.2.

Apart from the positivist stance expressed above in the formulation of (5.11), we will use an initial a posteriori progress estimation, $\hat{I}_0$, equal to 1. This means that we will be assuming full progress from the start and will let this indicator decay as the process advances.

On the other hand, we have not yet demonstrated that our criterion converges and, therefore, there is no theoretical guarantee of the optimization process stopping. This implies that we have to set a maximum limit on the number of iterations, $t_{\max}$, as a safety measure.

The remaining issue is the choice of the process noise covariance R, which, in our case, represents the degree of system inertia. As this is the only free parameter of the criterion, an incorrect choice could lead to undesired behavior. In the next section we show how criterion performance and robustness vary for different values of R.

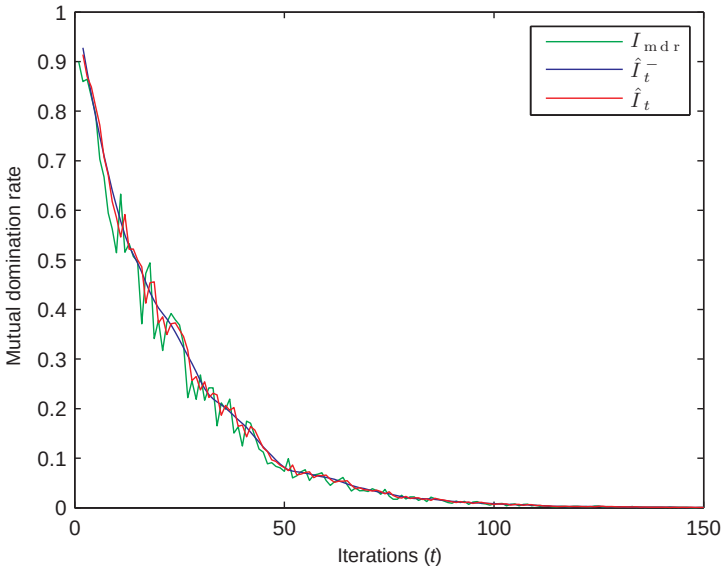


Figure 5.1: Evolution of the MDR indicator, $I_{\mathrm{mdr}}\left(\mathcal{P}_t^*, \mathcal{P}_{t-1}^*\right)$, and the a priori and a posteriori estimations $\hat{I}_t^-$ and $\hat{I}_t$ across iterations. Here the NSGA-II algorithm is supervised as it successfully solves the DTLZ3 problem (see Chapter 6 for more details).

Due to the particular assumptions enforced in the dynamic model, the values of Kalman gain K_t can be precomputed and stored in a table to speed up computation during the execution of the evolutionary algorithm.

Chapter 6 explores the issues presented here from an experimental point of view.

Initialize $t = 0$ and the a posteriori progress estimation $\hat{I}_0 = 1$.
Set R.
Set $t_{\max}$, the maximum number of iterations.
Set $\hat{I}_{\min}$, the minimum accepted value of the a posteriori estimation.
while $\hat{I}_t \geq \hat{I}_{\min}$ and $t < t_{\max}$ **do**
  Execute one iteration of the MOEA.
  $t = t + 1$.
  Compute the a priori progress estimation, $\hat{I}_t^-$, following (5.11).
  Calculate measured rate of improvement, z_t, as specified in (5.3) and (5.13).
  Determine the a posteriori estimation $\hat{I}_t$ from equations (5.12)-(5.16).
end while

Figure 5.2: Algorithmic description of the MGBM stopping criterion.

6

Experimenting with the MGBM Stopping Criterion

BEFORE diving into the comparison of the optimization algorithms presented in this thesis is first necessary to understand the viability of the application of the MGBM stooping criterion.

The viability of the proposal of Chapter 5 is established by comparing it with some other possible alternatives. In particular, it is compared with the relative versions of the hypervolume indicator and the additive epsilon indicator as progress indicators, and the application of statistical hypothesis testing to evidence assessment.

The theoretical and computational properties of the each of the components are discussed and contrasted. We also run a set of experimental tests. These tests are intended to assess each component combination under different circumstances in order to confirm that the method is capable of detecting "success" and "failure" stopping conditions. In these experiments we address some community-accepted test problems with the elitist non-dominated sorting genetic algorithm (NSGA-II) (Deb et al., 2002), the improved strength Pareto evolutionary algorithm (SPEA2) (Zitzler et al., 2002a) and the Pareto envelope-based selection algorithm (PESA) (Corne et al., 2000).

In order to establish the validity of MGBM we proposed two experiments. One is designed to gain a proper understanding of the properties of MBGM and its controlling parameter. The other compares MBGM with similar approaches with the aim of testing its validity and viability.

In particular we will present the results of applying the previously listed algorithms to solve three scalable multi-objective test problems —DTLZ3, DTLZ6 and DTLZ7— under different initial conditions. The choice of initial conditions is intended to bias the algorithm a priori so that we can test whether our criterion can resolve all possible target situations, i.e. either success or failure. Table summarize

6.1 Experimental Setup

As both experiments address the same test problems using the same MOEAs under the same biasing conditions, we will now describe the shared characteristics of both experiments.

Table 6.1: Parameters of the algorithms used in the experiments.

Common parameters	
Population size	100
NSGA-II	
Crossover probability	0.7
Distribution index for SBX	15
Mutation probability	0.1
Dist. index for polynomial mutation	20
SPEA2	
Crossover probability	0.7
Distribution index for SBX	15
Mutation probability	0.1
Dist. index for polynomial mutation	15
Ratio of sizes of population and archive	4:1
PESA	
Crossover probability	0.8
Distribution index for SBX	15
Mutation probability	0.1
Dist. index for polynomial mutation	15
Ratio of sizes of population and archive	4:1

6.2 Biasing the Optimization Outcome

As mentioned in Section 2.11 and Chapter 5, a stopping criterion should spot the conditions in which the execution of its associated algorithm should be terminated because it was either successful or failed to reach any solutions. To explore how good the stopping criteria is at doing this, the experiment parameters should be configured in such a way that the outcome of the optimization process is a priori biased towards a success or a failure.

For the success-biased experiments, we used three-dimensional problems ($M = 3$). The population size was set to 100 elements and the algorithms were left to run for 500 iterations. For the sake of reproducibility and to compare results, the internal parameters that we used in this study were the same as the values described by Khare (2002) (see Table 6.1). In that paper and in the preliminary exploratory experiments that we ran, this configuration was shown to correctly solve the three problems under study.

For failure biasing, the problems were configured with ten objectives ($M = 10$) and the other experimental conditions were unchanged. A series of experimental studies, including (Khare et al., 2003; Knowles and Corne, 2007; Purshouse and Fleming, 2007) and Deb (2001, pp.414–419), showed that there is an exponential dependence between the dimension of the objective space and the population size required to solve the problem correctly. When this ratio is not met because the population is smaller than it should be, then most of the population becomes non-dominated, and dominance-based ranking becomes useless, as it

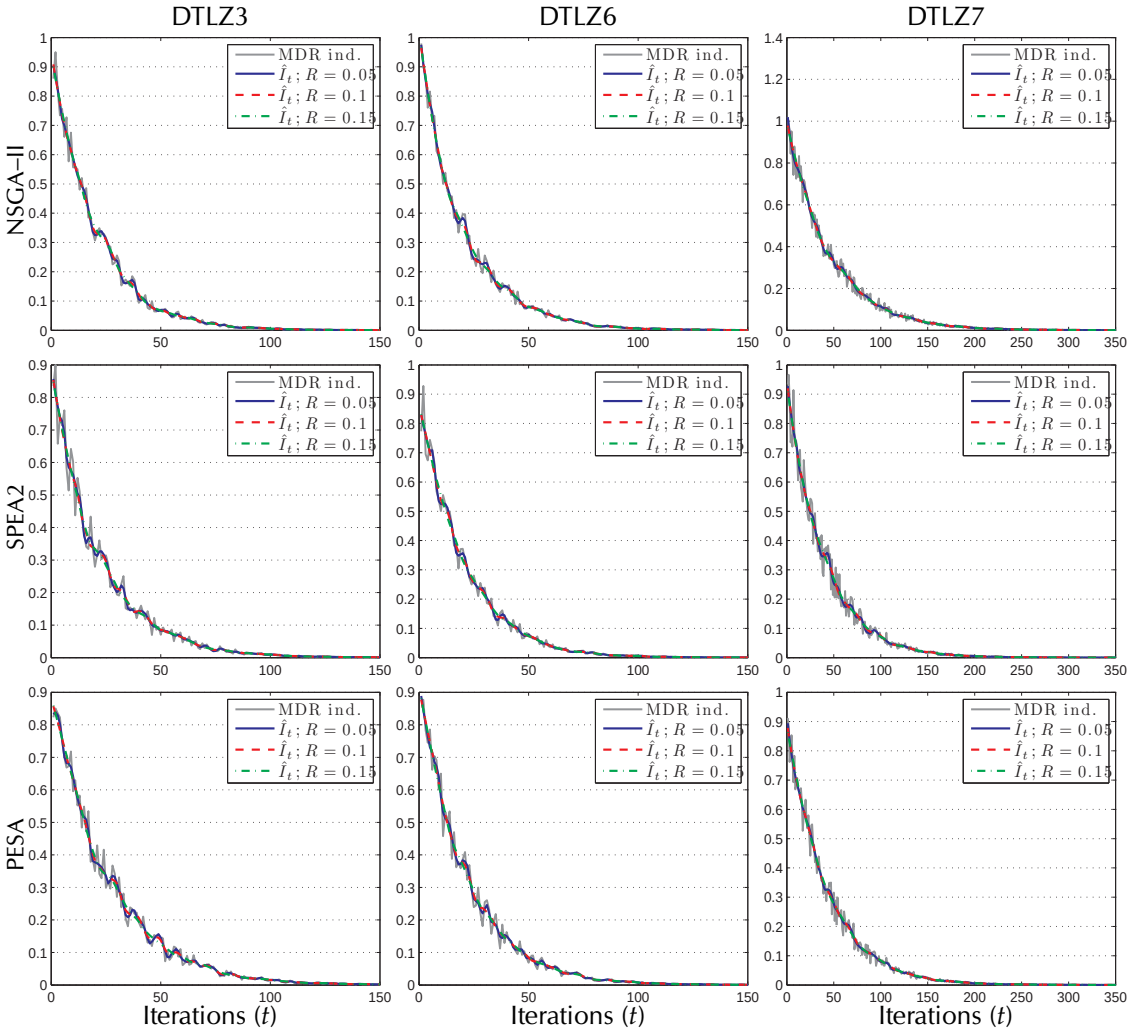


Figure 6.1: Mean values after each algorithm' iterations of the MDR progress indicator and the a posteriori estimation of progress, $\hat{I}_t$, for different values of R in a biased success experimental setup.

is unable to guide the search. The selected population size/number of objectives ratio has been shown to exhibit this behavior (Ishibuchi et al., 2008). Furthermore, some preliminary experiments were carried out in order to corroborate this point.

Experiments were carried out within the PISA (Bleuler et al., 2003) framework. An Intel Quad Core 3.4 GHz personal computer with 4 GB of RAM memory running the Linux operating system was used. The results reported here were output after 30 independent runs of the algorithms solving each of the problems.

6.3 Understanding MGBM Parameters

The purpose of this experiment was to explore the processes that take place under the hood of MGBM. We were particularly interested in observing how the Kalman filter keeps track of the evidence of progress provided by the indicator. Similarly, we wanted to study the impact

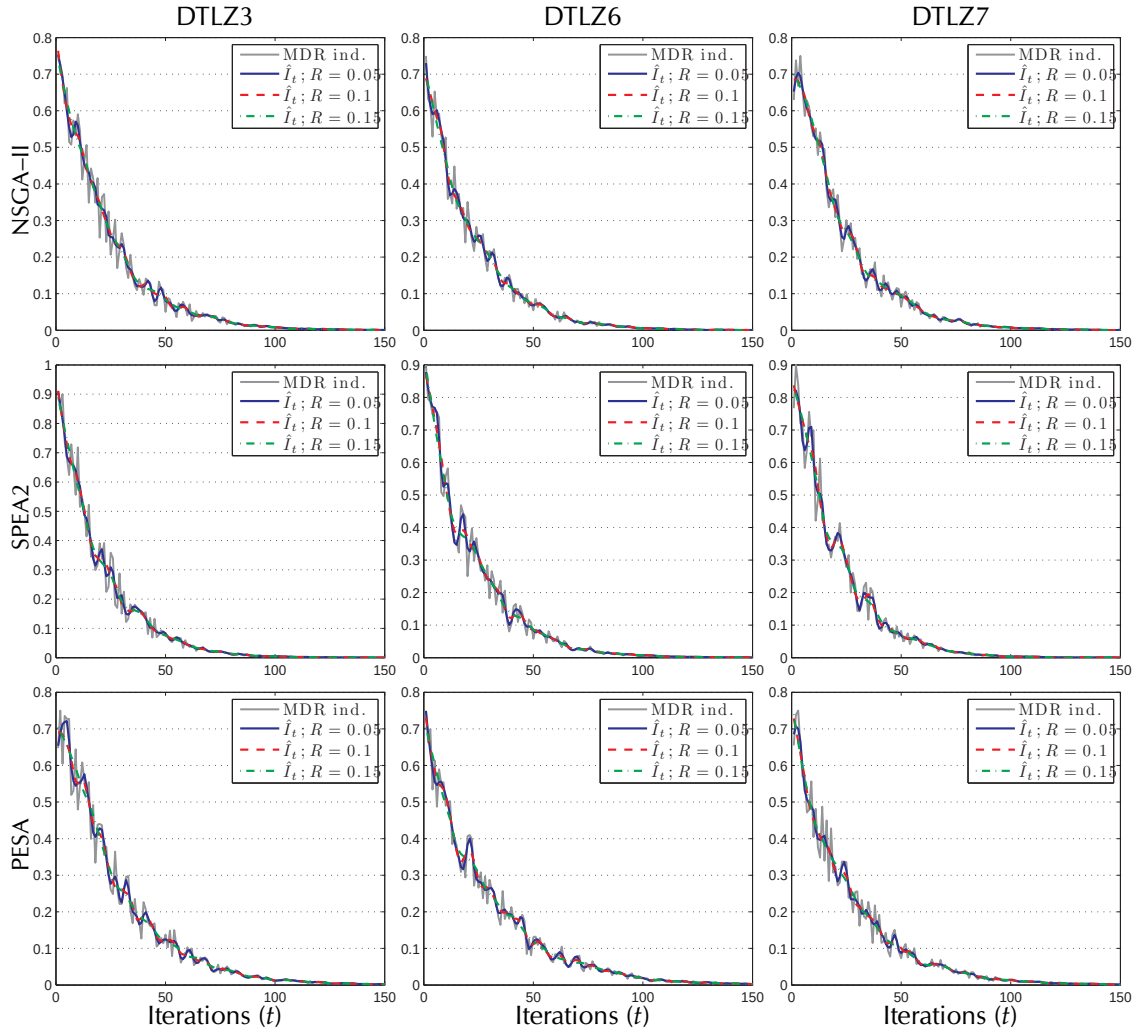


Figure 6.2: Evolution as iterations advance of the MDR progress indicator and the a posteriori estimation of progress, $\hat{I}_t$, for different values of R in a biased failure experimental setup. The plots represent the mean values of the quantities involved.

of R, the free parameter of the criterion. For this purpose, we applied MGBM with different values of R, in particular, $R = 0.05$, $R = 0.1$ and $R = 0.15$.

Figure 6.1 summarizes the evolution of the a posteriori estimators, $\hat{I}_t$, output when analyzing the execution of the three MOEAs solving the three test problems under study. In all cases the stopping condition was met when the algorithms became stable and their solutions were as close to the Pareto-optimal front as they would be at later generations. The criterion response was quicker or more inertial depending on the value of R. In the following experiment the performance indicator values of the solutions were measured at the iterations marked by the criterion. We found that the indicator values derived from the criterion are similar to the values output at the later algorithm execution stages.

Note that, in similar tests performed by Khare (2002), the algorithms were left to run for more iterations than suggested by the criterion. A set of analogous experiments were performed by Deb et al. (2004) with the same population size as ours but with unspecified

Table 6.2: Stop iterations suggested by the MGBM criterion with different values of R and the number of iterations used by Deb et al. (2004) and Khare (2002) when solving DTLZ3, DTLZ6 and DTLZ7 with similar configurations of NSGA-II, SPEA2 and PESA.

MOEA	MGBM			Deb et al. (2004)	Khare (2002)
	$R = 0.05$	$R = 0.1$	$R = 0.15$		
DTLZ3					
NSGA-II	91	104	115	500	500
SPEA2	121	132	149	500	500
PESA	125	129	138	500	500
DTLZ6					
NSGA-II	104	106	140	500	500
SPEA2	71	78	123	500	500
PESA	95	103	151	500	500
DTLZ7					
NSGA-II	237	259	275	200	—
SPEA2	269	305	330	200	—
PESA	279	298	326	200	—

internal parameters. For DTLZ3 and DTLZ6, our criterion also suggested halting the optimization with fewer iterations than they used. However, in the case of DTLZ7, the criterion suggested keeping the processes running for a longer than used in the above research, indicating that further processing was needed to reach the optimum. These results are summarized in Table 6.2.

Although the criterion appears to signal the algorithm to stop iterating earlier than in previous tests, these results raise a logical question. Are the solutions output at the iterations where the algorithm was stopped as good as the solutions output at the end of the simulation? Table 6.3 summarizes the mean hypervolume and additive epsilon indicator values measured at the iterations where the criterion fired and at the end of each execution of the algorithm. It indicates that there is no substantial difference between the quality of the solutions in the iterations selected by the criterion and the final iterations of the algorithms. One interesting feature is that the additive epsilon indicator values appear to be more homogeneous than the hypervolume values. This can be attributed to the fact that in later stages of the execution the solutions are improved in terms of diversity. This improvement is better captured by the hypervolume indicator.

The failure-biased experiments (see Figure 6.2) complement the above results. Again the difference between Kalman filters and the statistical hypothesis test is substantial.

6.4 MGBM and Its Possible Alternatives

In order to establish the validity and viability of MGBM, we proposed an experiment that contrasts MBGM with possible alternative approaches. The set of binary quality indicators

Table 6.3: Mean values of the quality indicators comparing the Pareto-optimal front, $\mathcal{O}^*$, and the non-dominated solutions front, $\mathcal{PF}^*$, of the problems output when the MGBM criterion was met and in the final iteration reported by Deb et al. (2004) and Khare (2002). Standard deviations of the values are shown in parentheses.

Stopping criteria	Hypervolume indicator	Additive epsilon indicator
	DTLZ3	
MGBM, $R = 0.05$	4.92×10^{-3} $_{(4.16\times10^{-4})}$	6.72×10^{-4} $_{(6.78\times10^{-5})}$
MGBM, $R = 0.10$	4.89×10^{-3} $_{(1.84\times10^{-4})}$	6.82×10^{-4} $_{(3.78\times10^{-5})}$
MGBM, $R = 0.15$	4.37×10^{-3} $_{(2.45\times10^{-4})}$	6.63×10^{-4} $_{(7.86\times10^{-5})}$
At $t = 500$	4.34×10^{-3} $_{(7.53\times10^{-4})}$	6.49×10^{-4} $_{(2.56\times10^{-5})}$
	DTLZ6	
MGBM, $R = 0.05$	5.90×10^{-3} $_{(3.35\times10^{-4})}$	8.72×10^{-4} $_{(7.82\times10^{-5})}$
MGBM, $R = 0.10$	5.91×10^{-3} $_{(1.62\times10^{-4})}$	8.60×10^{-4} $_{(6.77\times10^{-5})}$
MGBM, $R = 0.15$	5.82×10^{-3} $_{(4.16\times10^{-4})}$	8.37×10^{-4} $_{(4.46\times10^{-5})}$
At $t = 500$	5.76×10^{-3} $_{(3.73\times10^{-4})}$	8.15×10^{-4} $_{(4.97\times10^{-5})}$
	DTLZ7	
MGBM, $R = 0.05$	6.76×10^{-3} $_{(9.89\times10^{-5})}$	9.02×10^{-4} $_{(3.20\times10^{-5})}$
MGBM, $R = 0.10$	6.79×10^{-3} $_{(2.77\times10^{-4})}$	8.78×10^{-4} $_{(6.00\times10^{-5})}$
MGBM, $R = 0.15$	6.69×10^{-3} $_{(6.94\times10^{-4})}$	8.13×10^{-4} $_{(7.36\times10^{-5})}$
At $t = 200$	6.67×10^{-3} $_{(4.23\times10^{-4})}$	8.09×10^{-4} $_{(4.72\times10^{-5})}$

formulated for MOP solvers includes possible alternatives to the MDR indicator. We have chosen two popular indicators that were described in Section 2: the binary forms of the hypervolume and additive epsilon indicators (Knowles et al., 2006a).

Similarly, we chose a statistical hypothesis test scheme for comparison with the Kalman-based approach. In our case we have taken an approach based on the online convergence detection (OCD) method (Trautmann et al., 2009; Wagner et al., 2009). This method applies a one-sided χ^2 test (Chernoff and E. L., 1954) to determine if the variance of the measured indicator is below a set threshold, and a two-sided t-test (Gosset a.k.a. Student, 1908) to establish the linear trend of the indicator values. In our experiments we conformed to the parameter setup suggested by the OCD authors, that is, an iteration window of size 10, a variance threshold of -10^3 and a significance level of 0.05 for the statistical tests.

To make the study as comprehensive as possible, the components of MGBM and its alternatives were shuffled in all possible combinations. In other words, every progress indicator was tested with both evidence-gathering approaches. The results reported here were obtained after 30 independent runs of the algorithms solving each of the problems. The value of the MGBM parameter R was set to 0.1. The hypothesis test was conducted from a sample consisting of 50 consecutive iterations with a confidence of 95%.

Figures 6.3 and 6.4 show the performance of the different criteria in the success-biased and failure-biased experiments, respectively. The points in time where each criterion suggested stopping are marked with a circle.

Table 6.4: Quality indicator mean values comparing the Pareto-optimal front, $\mathcal{O}^*$, and the non-dominated solutions front, $\mathcal{PF}^*$, of the problems output when the criteria were met. Standard deviations of the values are shown in parentheses.

Stopping criteria	Hypervolume indicator	Additive epsilon indicator
	DTLZ3	
I_{mdr} + Kalman	4.89×10^{-3} $_{(1.84\times10^{-4})}$	6.82×10^{-4} $_{(3.78\times10^{-5})}$
I_{mdr} + hyp. test	4.39×10^{-3} $_{(3.84\times10^{-5})}$	6.49×10^{-4} $_{(1.80\times10^{-5})}$
I_{hyp} + Kalman	4.35×10^{-3} $_{(1.85\times10^{-4})}$	6.94×10^{-4} $_{(3.02\times10^{-5})}$
I_{hyp} + hyp. test	4.37×10^{-3} $_{(2.93\times10^{-4})}$	6.53×10^{-4} $_{(8.46\times10^{-5})}$
$I_{\epsilon+}$ + Kalman	4.53×10^{-3} $_{(2.15\times10^{-4})}$	6.48×10^{-4} $_{(3.42\times10^{-5})}$
$I_{\epsilon+}$ + hyp. test	4.28×10^{-3} $_{(3.01\times10^{-4})}$	6.45×10^{-4} $_{(4.28\times10^{-5})}$
	DTLZ6	
I_{mdr} + Kalman	5.91×10^{-3} $_{(1.62\times10^{-4})}$	8.60×10^{-4} $_{(6.77\times10^{-5})}$
I_{mdr} + hyp. test	5.78×10^{-3} $_{(2.55\times10^{-5})}$	8.23×10^{-4} $_{(1.32\times10^{-5})}$
I_{hyp} + Kalman	5.92×10^{-3} $_{(6.66\times10^{-5})}$	8.58×10^{-3} $_{(8.01\times10^{-5})}$
I_{hyp} + hyp. test	5.76×10^{-3} $_{(1.26\times10^{-4})}$	8.25×10^{-4} $_{(9.15\times10^{-5})}$
$I_{\epsilon+}$ + Kalman	5.97×10^{-3} $_{(2.45\times10^{-4})}$	8.62×10^{-4} $_{(2.75\times10^{-5})}$
$I_{\epsilon+}$ + hyp. test	5.72×10^{-3} $_{(9.05\times10^{-5})}$	9.19×10^{-4} $_{(4.73\times10^{-5})}$
	DTLZ7	
I_{mdr} + Kalman	6.79×10^{-3} $_{(2.77\times10^{-4})}$	8.78×10^{-4} $_{(6.00\times10^{-5})}$
I_{mdr} + hyp. test	6.10×10^{-3} $_{(4.99\times10^{-4})}$	7.94×10^{-4} $_{(2.69\times10^{-5})}$
I_{hyp} + Kalman	7.03×10^{-3} $_{(4.80\times10^{-4})}$	9.97×10^{-4} $_{(7.03\times10^{-5})}$
I_{hyp} + hyp. test	6.12×10^{-3} $_{(1.96\times10^{-4})}$	8.89×10^{-4} $_{(1.04\times10^{-4})}$
$I_{\epsilon+}$ + Kalman	6.82×10^{-3} $_{(1.96\times10^{-4})}$	9.49×10^{-4} $_{(1.04\times10^{-4})}$
$I_{\epsilon+}$ + hyp. test	6.57×10^{-3} $_{(1.08\times10^{-4})}$	7.82×10^{-4} $_{(1.31\times10^{-5})}$

The first, success-biased, case prompts one key conclusion: evidence gathering via Kalman filters is able to detect the optimizer stagnation at earlier stages than the statistical hypothesis test. This is because Kalman filters only require measurement-wise decisions, although they do take into account previous measurements, while hypothesis tests must analyze a relatively large sample of measurements. This difference also implies that Kalman filters require less computation to produce their results. This difference could perhaps be narrowed by reformulating the hypothesis test in a recursive form.

On the other hand, regardless of the evidence tracker used, the application of the MDR indicator is able to signal the algorithm to stop executing before the other alternatives. This raises the question of whether MDR makes the criterion activate before the optimization process has actually stopped. In order to clarify this issue we measured the hypervolume and additive epsilon indicator values comparing the problem Pareto-optimal set $\mathcal{F}^*$ and the non-dominated set output when the corresponding criterion was met, $\mathcal{P}^*$. These results are summarized in Table 6.4. We can safely say here that the indicator values yielded by the criteria are similar and adequate.

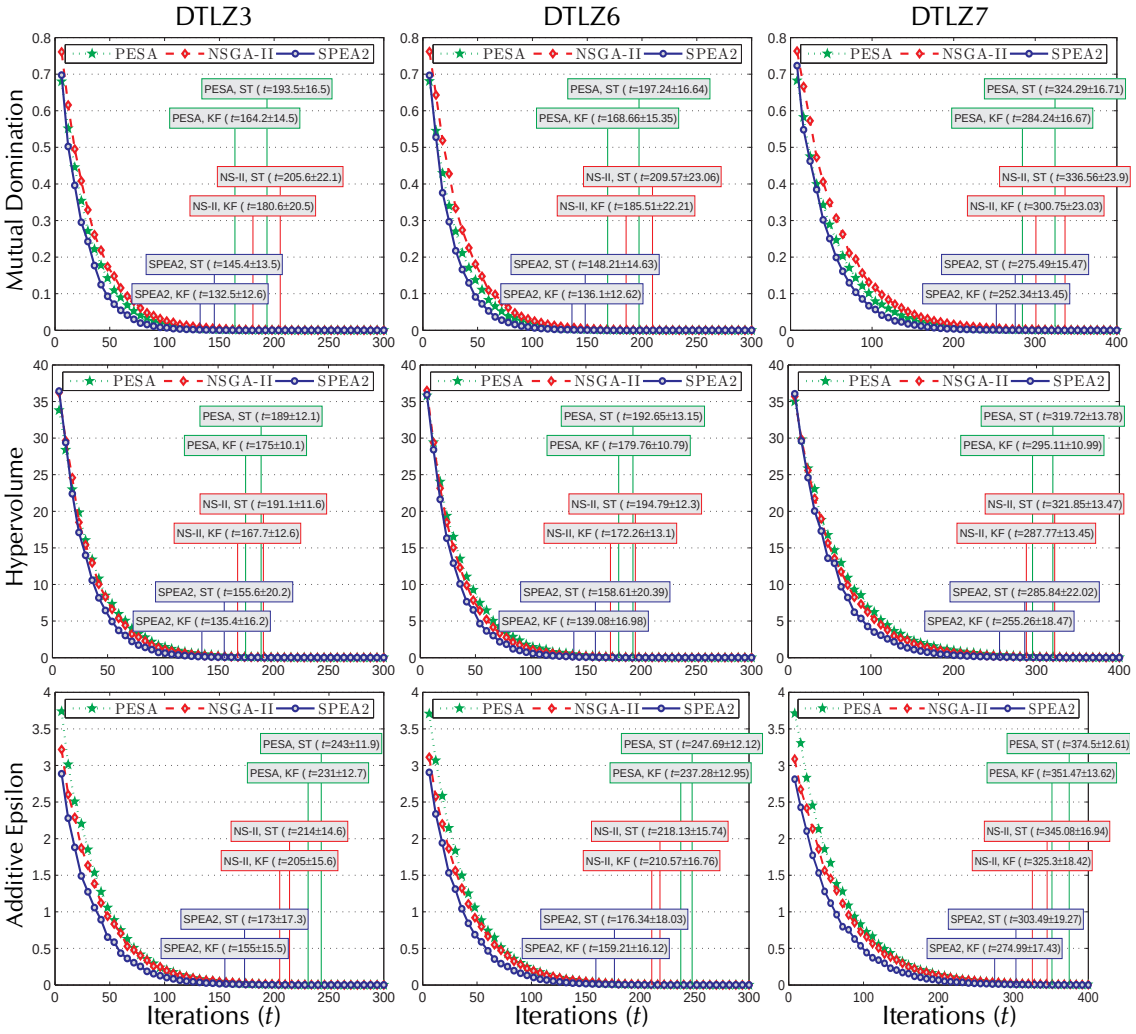


Figure 6.3: Evolution of the progress indicators mean values with iterations when solving problems biased towards success. Evidence gathering is performed via Kalman filter (KF) or statistical hypothesis test (ST). The mean stopping iteration suggested by each method is marked.

Clearly if the MGBM criterion were able to signal a stop condition earlier than the other variants but the solutions output at that iteration were of a lesser quality, the criterion would be useless. Table 6.4 shows that there is no substantial difference between the mean values of the quality indicators yielded by the different criteria. It can be inferred that, even though MGBM did fire earlier, it did not fire when the optimization was still in progress.

The early stopping indicated by MDR can be attributed to the fact that the other indicators, especially the hypervolume indicator, take into account the potential diversification in the non-dominated front of the algorithms. This is relatively less important when dealing with many-objective problems like the ones discussed above. The main concern in this class of application is to get as close as possible to the Pareto-optimal front, as the spread of solutions would place an even greater demand on computational resources.

The failure-biased experiments (Figure 6.4) complement the results described above. The above rationale cannot be extrapolated as-is to the failure-biased experiments. Note that this

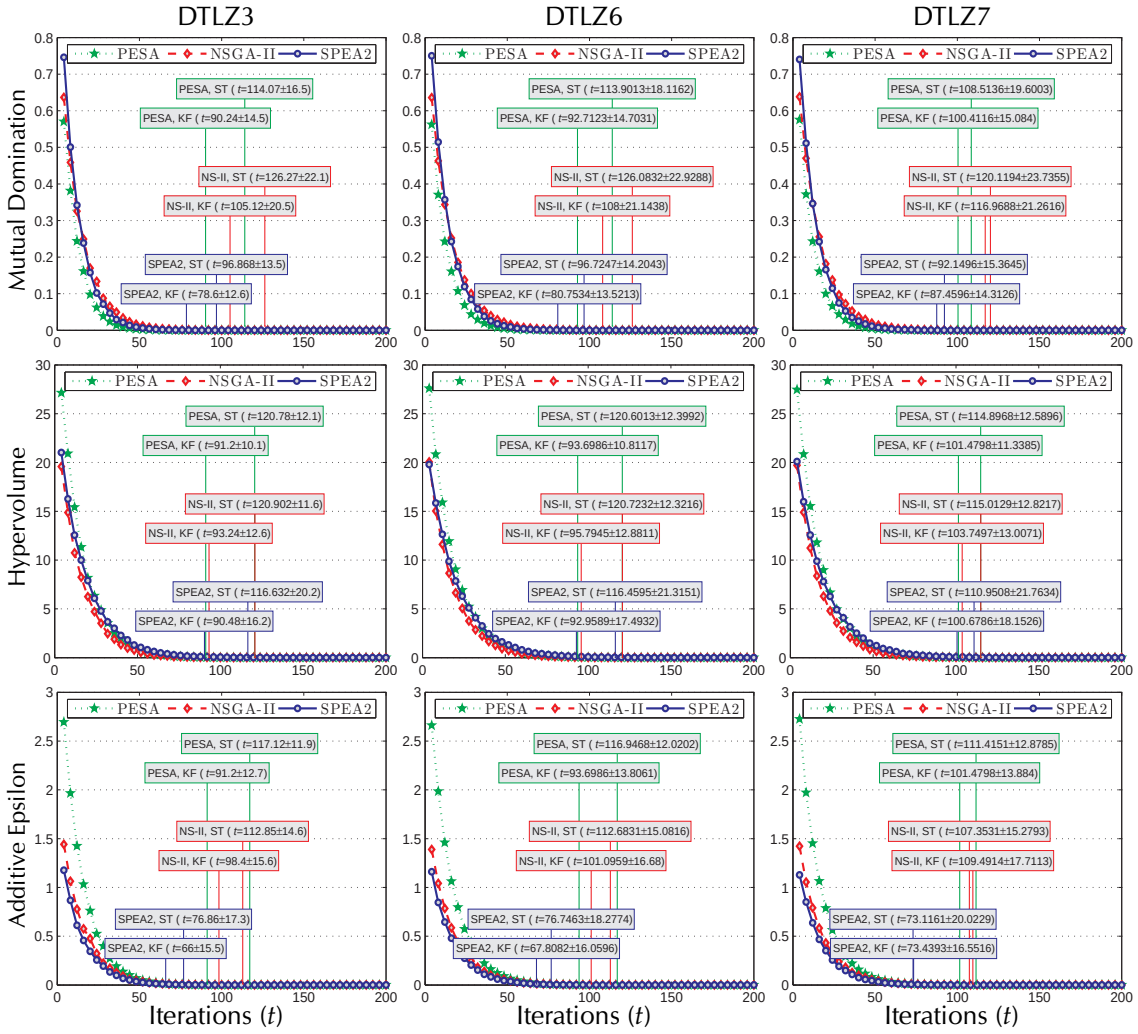


Figure 6.4: Evolution of the progress indicators mean values with iterations when solving problems biased towards failure. Evidence gathering is performed via Kalman filter (KF) or statistical hypothesis test (ST). The mean stopping iteration suggested by each method is marked.

class of experiment has not been previously proposed elsewhere, although we think it is indispensable to gain a complete experimental understanding of the matter. It is probably conceptually impossible to determine if the criteria actually managed to detect this situation. What we did find is that all the criteria did signal the algorithm to stop, and none left the algorithm running for an indefinite (and possibly infinite) number of iterations.

Still, there are some noteworthy points. It is again noticeable that the Kalman filter is able to detect the non-progress condition earlier. However, this difference is not as big as in the success-biased experiments. Similarly, the fact that all the evidence gathered performed more or less the same, regardless of the indicator used, is very illustrative. This is particularly noticeable in the case of the DTLZ7 problem.

The above results raise another question. What is the resource consumption of each of the combinations? This property is usually reported as the time taken by the process to terminate. However, this approach is hard to reproduce. For this reason, we will measure the number

of CPU operations carried out to process the stopping criterion in each iteration instead. We found this approach to be sounder and more easily reproducible than just measuring the duration. To do this, we employed the OProfile software profiling tool (Levon, 2004). The profiling tool was configured in such way that it only reported the CPU operations run by the processes of interest.

The mean number of CPU operations run by each evidence gatherer and progress indicator combination are summarized in Table 6.5. We find that MGBM (the combination of MDR and Kalman filter) runs the fewest operations of all in most cases. In all cases, the application of the statistical hypothesis test is far more computationally expensive than its Kalman counterpart. This is due to the simplicity of the operations carried out by the Kalman filter compared to the repetitive assessment of indicator values of previous iterations used by the statistical hypothesis test. Also notable is the increment in the amount of computation of the hypervolume indicator, particularly evident in the failure-biased experiments. This can be attributed to its exponential relation to the number of objectives. Last but not least, let us look at the homogeneity of the results, a point that lends support to the possibility of outputting similar performances when dealing with other problems of the same magnitude.

6.5 Remarks

In this chapter and the previous one we have presented a novel stopping criterion to be used in multi-objective optimization problems. In particular, we proposed a global stopping criterion, named MGBM criterion, which combines the mutual domination rate (MDR) improvement indicator with a simplified Kalman filter that is used in the evidence-gathering process. The MDR indicator is a special-purpose solution designed for the stopping task. It is capable of gauging the progress of the optimization with a low computational cost and is therefore suitable for solving complex or many-objective problems. Although the stopping criterion issue apparently plays a secondary role, real-world practical experiences underscore its importance.

We have described the criterion theoretically and have examined its performance on some test problems. It was also compared with similar approaches to the issue. From these experiments we have found that MGBM is a good starting point for research in this direction. Obviously, more experimentation is required, and other types of filters must be tested. Research on creating the necessary assessment tools to be able to gauge the performance of the criteria would perhaps not go amiss either. It should be noted, however, that the criterion has been successfully applied by the authors in a series of studies that deal with high-dimensionality multi-objective problems (Martí et al., 2008a,b,c, 2009b).

A salient issue is the interpretation of the final algorithm state in order to establish the reason for the process being stopped. Evidence gathered during this research indicates that some conclusions can be drawn on this point by analyzing the number of dominated and non-dominated individuals in the population. These results are consistent with outcomes previously presented by Ishibuchi et al. (2008); Khare et al. (2003); Knowles and Corne (2007); Praditwong and Yao (2007); Purshouse and Fleming (2007) and Deb (2001, pp. 414–419).

Another key issue is to capture the diversification process that takes place as part of

Table 6.5: Mean CPU operations per iteration performed by each stopping criterion for every experimental condition. Standard deviations of the values are shown in parentheses. Minimal values are highlighted in bold.

Evidence gatherers	**Progress indicators** MGBM	Hypervolume	Additive epsilon
	— Success-biased —		
	DTLZ3		
Kalman	$\mathbf{2.877 \times 10^3}$ (1.01×10^2)	2.074×10^4 (3.71×10^2)	3.016×10^3 (1.10×10^2)
Hyp. test	8.390×10^3 (1.19×10^2)	2.670×10^4 (8.30×10^2)	8.716×10^3 (1.78×10^2)
	DTLZ6		
Kalman	2.885×10^3 (1.32×10^2)	2.521×10^4 (3.47×10^2)	$\mathbf{2.879 \times 10^3}$ (1.08×10^2)
Hyp. test	8.286×10^3 (1.87×10^2)	2.380×10^4 (2.12×10^2)	8.496×10^3 (1.88×10^2)
	DTLZ7		
Kalman	$\mathbf{2.687 \times 10^3}$ (1.11×10^2)	2.238×10^4 (4.02×10^3)	2.858×10^3 (1.09×10^2)
Hyp. test	7.839×10^3 (1.37×10^2)	2.396×10^4 (3.718×10^3)	7.823×10^3 (1.75×10^2)
	— Failure-biased —		
	DTLZ3		
Kalman	$\mathbf{1.443 \times 10^4}$ (9.18×10^2)	3.839×10^9 (9.18×10^7)	1.492×10^4 (2.83×10^3)
Hyp. test	4.146×10^4 (1.00×10^3)	1.456×10^{10} (5.84×10^8)	4.884×10^4 (9.06×10^2)
	DTLZ6		
Kalman	$\mathbf{1.351 \times 10^4}$ (7.59×10^2)	1.548×10^{10} (8.21×10^8)	2.421×10^4 (3.47×10^3)
Hyp. test	3.859×10^4 (1.07×10^3)	1.108×10^{10} (7.49×10^8)	3.910×10^4 (2.01×10^3)
	DTLZ7		
Kalman	$\mathbf{1.418 \times 10^4}$ (1.18×10^3)	5.862×10^9 (1.91×10^8)	1.497×10^4 (1.74×10^3)
Hyp. test	4.163×10^4 (1.43×10^3)	1.003×10^{10} (8.98×10^8)	4.385×10^4 (9.78×10^2)

the optimization process. It has been documented that after the population hits a local Pareto front it starts exploring along that front. An indicator capable of measuring the degree to which the optimization algorithm is actively exploring the search space could perhaps improve the results presented here.

7

Verifying the Model-Building Hypothesis

> The great tragedy of Science: the slaying of a beautiful hypothesis by an ugly fact.
>
> *Thomas H. Huxley*

PRIOR to testing MONEDA as a whole is is necessary to verify the model-building GNG proposed in Chapter 3 and to identify if it deals properly with the model-building issue debated. Therefore, it is helpful to devise a comparative experiment that casts light on the performances of a selected set of model-building algorithms subject to the same conditions to deal with a group of complexity-scaling problems. It would be similar to the preliminary experiment presented in Chapter 3 but carrying out a broader and fuller set of tests.

In particular, we deal with a selection of the Walking Fish Group (WFG) continuous and scalable test problems set (Huband et al., 2005, 2006a).

A MOEDA framework is shared by the model-building algorithms involved in the tests in order to ensure the comparison and reproducibility of the results. Two well-known MOEAs, the non-dominated sorting genetic algorithm II (NSGA-II) (Deb et al., 2002) and the strength Pareto evolutionary algorithm (SPEA2) (Zitzler et al., 2002a), were also applied as a baseline for the comparison.

The model-building algorithms involved in the tests were:

- Bayesian networks, as used in MrBOA;
- randomized leader algorithm, k-means algorithm and E-M algorithm, as described for MIDEAs;
- $(1 + \lambda)$-CMA-ES as described in (Igel et al., 2007); and
- GNG and its model-building version, MB-GNG.

This assortment of algorithms offers a broad sample of different approaches, ranging from the most statistically rigorous algorithms, such as Bayesian networks, E-M or CMA-ES,

Table 7.1: Parameters of the algorithms used in the experiments.

Common parameters	
Population size (n_{pop})	$250 \cdot 10^{\frac{M}{3}-1}$
Shared EDA framework	
Selection percentile (α)	0.3
Substitution percentile (ω)	0.3
GNG and MB-GNG	
Number of initial GNG nodes (N_0)	2
Maximum edge age (v_{max})	40
Best node learning rate (ϵ_{b})	0.1
Neighbor nodes learning rate (ϵ_{v})	0.05
Insertion error decrement rate (δ_{I})	0.1
General error decrement rate (δ_{G})	0.1
Accumulated error threshold (ρ)	0.2
$\hat{P}_t$ to N_{max} ratio (γ)	0.5
Randomized leader algorithm	
Maximum number of clusters	$\lceil 0.5 \lfloor \alpha n_{\text{pop}} \rfloor \rceil$
Threshold for the leader algorithm	0.1
***k*-means algorithm**	
Number of clusters	$\lceil 0.25 \lfloor \tau n_{\text{pop}} \rfloor \rceil$
Stopping threshold	0.0001
Expectation maximization	
Maximum number of clusters	$\lceil 0.5 \lfloor \tau n_{\text{pop}} \rfloor \rceil$
Threshold for the leader algorithm	0.1

Bayesian networks	
Number of parents or a variable	5
Number of mixture components	3
Threshold of leader algorithm	0.1
Covariance matrix adaptation	
Offspring number (λ)	1
Target success probability ($p_{\text{succ}}^{\text{target}}$)	$\frac{1}{5+\sqrt{\frac{\lambda}{2}}}$
Step size damping (d)	$1 + \frac{n}{2\lambda}$
Success rate averaging parameter (c_p)	$\frac{\lambda p_{\text{succ}}^{\text{target}}}{2+p_{\text{succ}}^{\text{target}}}$
Cumulation time horizon parameter (c_c)	$\frac{2}{n+2}$
Covariance matrix learning rate (c_{cov})	$\frac{2}{n^2+6}$
Success rate threshold (p_{thresh})	0.44
NSGA-II	
Crossover probability (p_c)	0.7
Distribution index for SBX (η_c)	15
Mutation probability (p_m)	$\frac{1}{n_{\text{pop}}}$
Dist. index for polynomial mut. (η_m)	20
SPEA2	
Crossover probability (p_c)	0.7
Distribution index for SBX (η_c)	15
Mutation probability (p_m)	$\frac{1}{n_{\text{pop}}}$
Dist. index for polynomial mut. (η_m)	20
Ratio of pop. to archive sizes	4 : 1

to others, like the leader algorithm and MB-GNG, that have some clear shortcomings in the context of their original application scope. Nevertheless, they can also be assumed to deal with outlying elements in a more adequate manner.

The parameters of the different algorithms involved in the experiments are summarized in Table 7.1.

The shared MOEDA framework of introduced in Chapter 3 with be used here as described there.

7.1 Experimental Setup

The problems to be addressed are part of the Walking Fish Group problem toolkit (WFG) (Huband et al., 2006b). This is a toolkit for creating complex synthetic multi-objective test problems that can be devised to exhibit a given set of target features. Appendix C contains a description of these problems.

From the set of nine problems, the test functions WFG4 to WFG9 were selected because of the simple form of their Pareto-optimal fronts, which lie on the first orthant of a unit hypersphere. For this reason, the progress of the optimization process can be determined

without having a sampled version of the Pareto-optimal front. In particular, we measure the similarity of the current non-dominated front, $\mathcal{PF}_t^*$, to the Pareto-optimal front as the mean distance of the elements of $\mathcal{PF}_t^*$ to the origin of coordinates minus one,

$$I_{\text{prog}} = \frac{\sum_{\boldsymbol{x}\in\mathcal{F}_t^*}\left(\sum_{m=1}^{M}\left(f_m(\boldsymbol{x})-1\right)^2\right)^{0.5}}{|\mathcal{F}^*|}\,. \tag{7.1}$$

For this reason, the local progress of the algorithms can be easily determined as executions taking place without having to turn to more computationally expensive options such as performance indicators.

Even so, assessing the progress of the algorithms in high dimensions is a complicated matter. To do this, we used the MGBM multi-objective optimization cumulative stopping criterion introduced in previous chapters. This criterion combines the measurement of progress across iterations I_{prog} with a simplified Kalman filter that is used for the evidence-gathering process.

Performance indicators are required to gauge and compare the quality of the solutions yielded by each algorithm. In these experiments the hypervolume indicator (Knowles et al., 2006a) was used for performance. This indicator gauges how similar the solution yielded by each algorithm is to the Pareto-optimal front of the problem. Therefore, it requires an explicit sampling of that front, which is not viable in problems with many objectives. To address this issue, we took an approach similar to the method adopted by the purity performance indicator (Bandyopadhyay et al., 2004; Ishibuchi and Murata, 1998). A combined set $\mathcal{PF}^+$ is defined as the union of the solutions obtained from the different algorithms across all the experiment executions. $\tilde{\mathcal{O}}^*$ is then determined by extracting the non-dominated elements,

$$\boldsymbol{x} \in \tilde{\mathcal{O}}^* \text{ iff } \boldsymbol{x} \in \mathcal{PF}^+ \text{ and } \nexists \boldsymbol{y} \in \mathcal{PF}^+ \text{ such that } \boldsymbol{y} \prec \boldsymbol{x}\,. \tag{7.2}$$

Although this procedure circumvents the problems of performing a direct sampling of the Pareto-optimal front shape function, special precautions should be taken when interpreting the results. Notice that the algorithm's performance will be measured with regard to the set of overall best solutions and not against the actual Pareto-optimal front. We consider this to be a valid approach, though, since the intention of these experiments is to compare the different model-building algorithms rather than actually solving the problems.

Each problem was configured with 3, 5, 7 and 9 objective functions. For all cases, the decision space dimension was set at 15. The experiments were carried out under the PISA experimental framework (Bleuler et al., 2003). All the algorithms were executed 30 times for each problem/dimension pair.

Statistical hypothesis tests have to be applied to validate the results of different executions. Different frameworks for carrying out this task have been already discussed by other authors (see for example (Coello Coello et al., 2007; Knowles et al., 2006a; Zitzler et al., 2008)).

In our case, we performed a Kruskal–Wallis test and the Conover–Inman procedure as described in Appendix B. A significance level, α, of 0.05 was used for all the tests.

Besides measuring how good the solutions output by the algorithms are, it is also very important to analyze how long it takes the algorithms to reach the solutions. For these experiments we measured two variables: the number of objective function evaluations and

the number of floating-point operations carried out by each model-building algorithm. This last measurement assumes that all floating-point operations have to do with the optimization process itself. This requirement can be easily met under experimental conditions. There are a number of profiling tools that are capable of tracking the number of floating-point operations that have taken place as part of a process. The details of how this task is carried out are provided in Section B.1. As the study also covered NSGA-II and SPEA2 and they do not perform model building, we measured the operations dedicated to the application of the evolutionary operators in their case.

7.2 Results

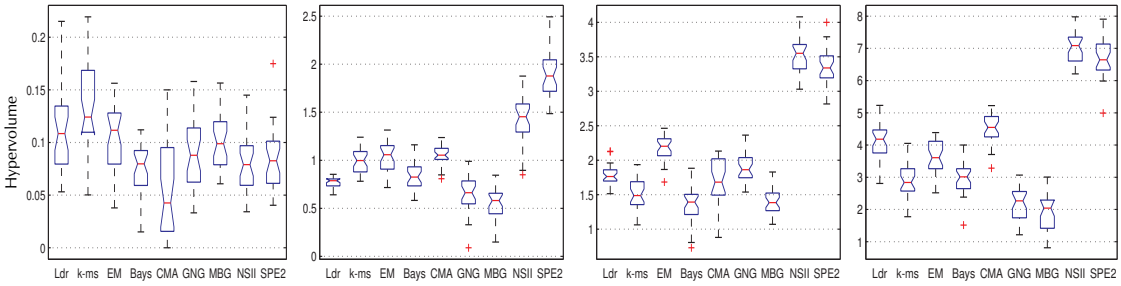
As already explained, the scope of the experiments reported here is to validate or reject the hypotheses stated in Chapter 3. For this reason, the performance of each algorithm is compared in terms of both the quality of the solutions that they generate and their cost in terms of computational resources. Particularly, we are concerned with the number of floating-point operations dedicated to the model-building task and with the number of function evaluations performed.

The first results have to do with the WFG4 problem. WFG4 is a separable and strongly multi-modal problem that, like the other problems, has a concave Pareto-optimal front. This front lies on the first orthant of a hypersphere of radius one located at the origin. The separability property should, in theory, allow Bayesian networks-based approaches to perform well, as already reported by Pelikan (2005b).

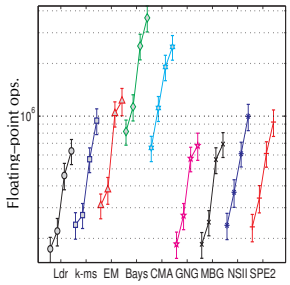
Figure 7.1 summarizes the outcome of the experiments related to this problem. These results show what will be a common characteristic of all the results presented here. In low dimensionality, in particular with $M = 3$, none of the models yielded substantially different results as illustrated in Figure 7.1a. Better results could possibly be achieved by further tuning the parameters. However, this situation gradually changes as the number of objectives increases (see Figures 7.1b–7.1d). In these cases, the least robust approaches (statistically speaking), such as the leader algorithm, GNG and MB-GNG, outperform the others in terms of approximation to the Pareto-optimal front, with the exception of the 7-objective case, where Bayesian networks outperform the other algorithms. This is a result that could be attributed to the fact that this is a separable problem. This outcome can be verified by looking at the statistical hypothesis test results shown in Figure 7.1g.

Another illustrative analysis emerges when analyzing the mean number of floating-point operations and the number of function evaluations shown in Figures 7.1e and 7.1f. Let us draw attention in the first figure to the fact that EM, Bayesian networks and CMA-ES consume far more resources and exhibit poorer scaling properties with regard to the other algorithms even with respect to the standard MOEAs used for a baseline comparison. The fact that such a rise in the computational demand of those algorithms did not lead to an increase in the number of function evaluations is even more interesting. Therefore, this increase in the computational cost was not caused by an increase in the amount of searching done; instead, it can be attributed to just the creation of the data models.

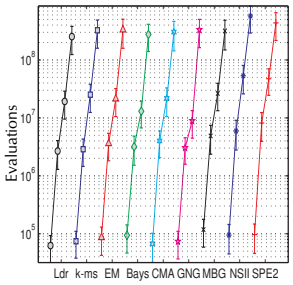
WFG5 is also a separable problem but it has a set of deceptive locally optimal fronts. This feature is meant to evaluate the capacity of the optimizers to avoid getting trapped in local



(a) $M = 3$ (b) $M = 5$ (c) $M = 7$ (d) $M = 9$



(e) Model–building CPU ops.



(f) Objective function evals.

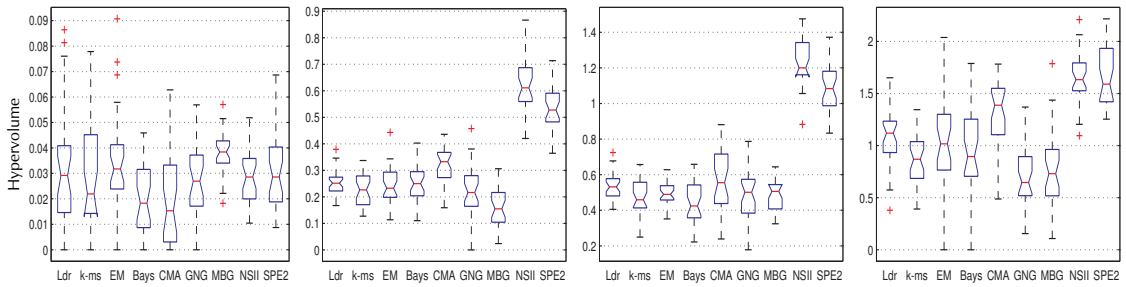
(g) Instances with statistically significant better results.

	Ldr	k-ms	EM	Bays	CMA	GNG	MBG	NSII	SPE2
Ldr	—	3,5	5,7	5	5,9	7		5,7,9	5,7,9
k-ms		—	7,9		5,7,9	7		5,7,9	5,7,9
EM			—		9			5,7,9	5,7,9
Bays				—	5,7,9	7	3	5,7,9	5,7,9
CMA					—	3,7	3	3,5,7,9	3,5,7,9
GNG						—		5,7,9	5,7,9
MBG							—	5,7,9	5,7,9
NSII								—	5
SPE2									—

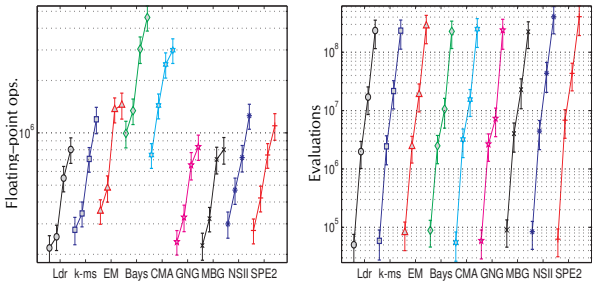
Figure 7.1: Results for problem WFG4 of applying for model–building the randomized leader algorithm (Ldr), the k–means (k-ms) algorithm, expectation maximization (EM), Bayesian networks (Bays), covariance matrix adaptation evolutionary strategy (CMA), growing neural gas network (GNG) and the model–building growing neural gas network (MBG). For comparison reasons NSGA–II (NSII) and SPEA2 (SPE2) evolutionary algorithms are also shown. Figs. (a)–(d) summarize the statistical description of the hypervolume values obtained after each experiment as box–plots. Figure (e) shows the progression across problem dimensions of the floating–point operations used by the model–building algorithms, while Figure (f) contain a similar representation but for the number of function evaluations. Table (g) summarizes the outcome of performing the statistical hypothesis tests. The numbers shown are the problem dimension where the test detected a statistically significant better indicator values of the algorithm in each row with respect of those in the columns.

optima. Figure 7.2 shows the results for this problem. In spite of the hurdle of the multiple local optima, the results are quite consistent with those obtained for WFG4. The scenario that differentiates the three-objective problem from the other dimensions is repeated here, save that CMA-ES is the algorithm that yields better solutions in the $M = 7$ case. In the other two "high" dimensions, 5 and 9, MB-GNG is the algorithm that yields the best results. As in WFG4, if we contrast the floating-point operations and the objective function evaluations, it is clear that EM, Bayesian networks and CMA-ES required much more computational time to perform a similar level of search space exploration.

The next problem, WFG6, is a separable problem without the strong multi-modality of WFG4. Figure 7.3 summarizes the comparative performances of the different algorithms when dealing with this problem. In this case, MB-GNG outperforms the other algorithms in terms of Pareto optimality in all the high-dimensional cases. It is also noticeable that Bayesian networks yield similar results to non-statistically rigorous algorithms. This can be attributed to problem separability. The pattern of floating-point operations and function evaluations relations already discussed in the previous problems is also present here.



(a) $M = 3$ (b) $M = 5$ (c) $M = 7$ (d) $M = 9$



(e) Model–building CPU ops. (f) Objective function evals.

(g) Instances with statistically significant better results.

	Ldr	k-ms	EM	Bays	CMA	GNG	MBG	NSII	SPE2
Ldr	—			5,9			3	5,7,9	5,7,9
k-ms		—		5,7,9			3	5,7,9	5,7,9
EM			—	5,9				5,7,9	5,7,9
Bays				—	5,7,9		3	3,5,7,9	3,5,7,9
CMA					—	3	3	3,5,7,9	3,5,7,9
GNG						—	3	5,7,9	5,7,9
MBG							—	5,7,9	5,7,9
NSII								—	
SPE2									—

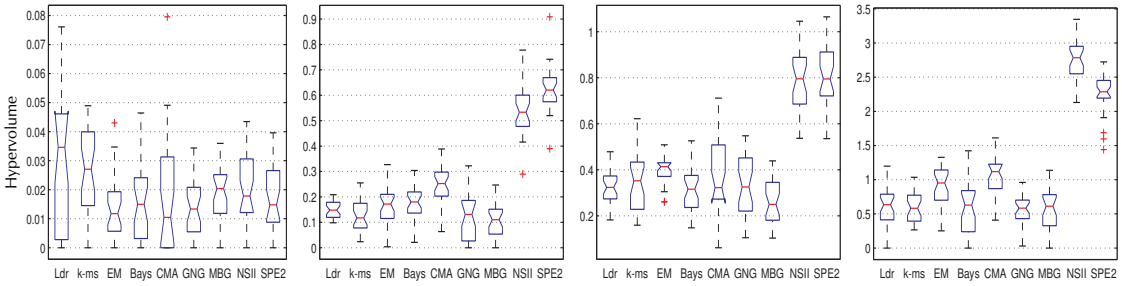
Figure 7.2: Results when solving the WFG5 problem. See Fig 7.1 for a description of each subfigure and abbreviations.

The remaining three problems have the added difficulty of having a parameter-based bias. WFG7 is uni-modal and separable, like WFG4 and WFG6. Its results are reported in Figure 7.4. In this case, GNG and MB-GNG outperform their peers in the problems with 5 and 7 objectives. However, Bayesian networks yielded better average results when tackling the problem with 9 objectives, although this improvement was not deemed as statistically significant.

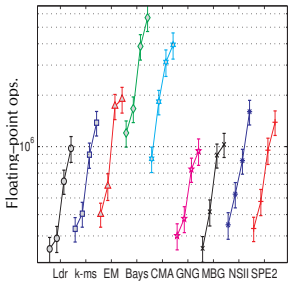
WFG8 is a non-separable problem and its results are illustrated in Figure 7.5. So far, this is the problem where non-rigorous algorithms most obviously outperformed the others with a more solid statistical foundation in the higher dimensionality (in objective function space). In the nine-objective case (Figure 7.5d) there seems to be little difference among the results of the leader algorithm, CMA-ES, GNG and MB-GNG. However, the much higher cost of running CMA-ES than the other three approaches is much clearer from the results shown in Figure 7.5e.

Finally, WFG9 is non-separable, multi-modal and has deceptive local optima. These properties make WFG9 the hardest problem of all the problems chosen for the study. Figure 7.6 shows the results obtained with the tested algorithms. As in the previous experiments, MB-GNG manages to yield the best results, in this case, sharing its success with the leader algorithm in the nine-objective case.

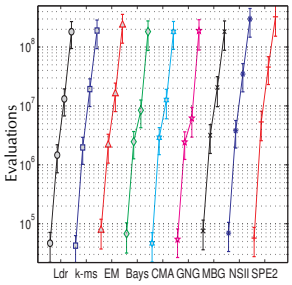
Looking at this relatively large set of results, even in the light of the most advantageous representation chosen, they are rather cumbersome. First of all, it is noticeable that there is no clear winner in the three- objective problems, where the different model-building algorithms alternately outperform each other. This changes as the number of objectives is



(a) $M = 3$ (b) $M = 5$ (c) $M = 7$ (d) $M = 9$



(e) Model–building CPU ops.



(f) Objective function evals.

(g) Instances with statistically significant better results.

	Ldr	k-ms	EM	Bays	CMA	GNG	MBG	NSII	SPE2
Ldr	—		7,9	5	5,9			5,7,9	5,7,9
k-ms		—	5,9	5	5,9			5,7,9	5,7,9
EM			—		5			5,7,9	5,7,9
Bays				—	5,9			5,7,9	5,7,9
CMA					—			5,7,9	5,7,9
GNG						—		5,7,9	5,7,9
MBG							—	5,7,9	5,7,9
NSII								—	5
SPE2									—

Figure 7.3: Results when solving the WFG6 problem. See Fig 7.1 for a description of each subfigure and abbreviations.

increased. Noticeably, model-building approaches that rely on solid statistical foundations, such as Bayesian networks, EM, or CMA-ES are outperformed by the others without such properties. In terms of computational cost, we find that, while the overall number of function evaluations remained within similar ranges for the different algorithms, the effort expended on model building was far greater for EM, CMA-ES and the Bayesian networks.

7.3 Analyzing the Results

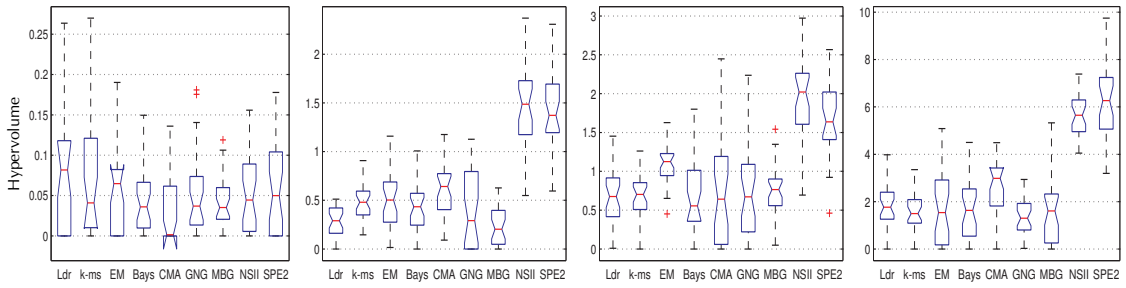
It is not easy to draw conclusions from the previous studies, as it implies cross-examining and comparing the results presented separately in Figures 7.1–7.6. For this reason, we decided to adopt a more integrative representation along the lines of the schema proposed in (Bader, 2010; Bader and Zitzler, 2011).

That is, for a given set of algorithms $A_1,\ldots,A_K$, a set of P test problem instances $\Phi_{1,m},\ldots,\Phi_{P,m}$, configured with m objectives, the function $\delta(\cdot)$ is defined as

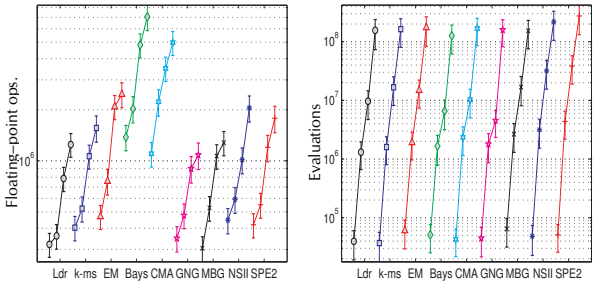
$$\delta\left(A_i, A_j, \Phi_{p,m}\right) = \begin{cases} 1 & \text{if } A_i \gg A_j \text{ solving } \Phi_{p,m} \\ 0 & \text{in other case} \end{cases}, \tag{7.3}$$

where the relation $A_i \gg A_j$ defines whether A_i is significantly better than A_j when solving the problem instance $\Phi_{p,m}$, as computed by the above statistical tests.

Relying on $\delta(\cdot)$, the performance index $P_{p,m}(A_i)$ of a given algorithm A_i when solving



(a) $M = 3$ (b) $M = 5$ (c) $M = 7$ (d) $M = 9$

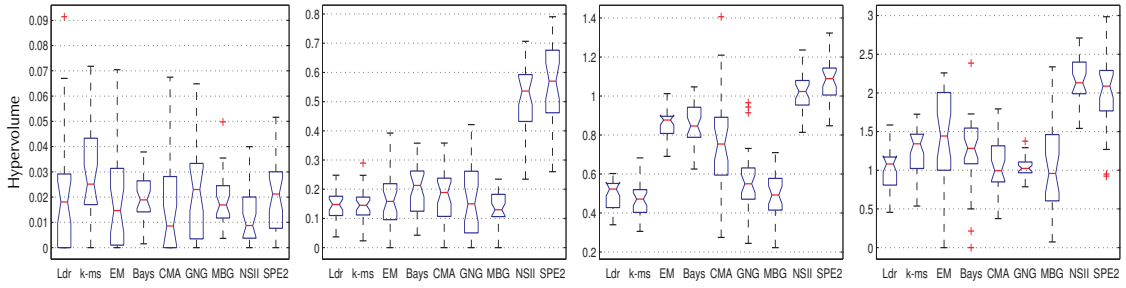


(e) Model–building CPU ops. (f) Objective function evals.

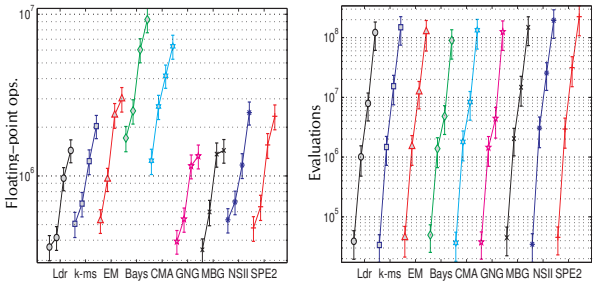
(g) Instances with statistically significant better results.

	Ldr	k-ms	EM	Bays	CMA	GNG	MBG	NSII	SPE2
Ldr	—	5	5,7	5	5,9			5,7,9	5,7,9
k-ms		—	7		5,9			5,7,9	5,7,9
EM			—		5,9			5,7,9	5,7,9
Bays				—	5,9			5,7,9	5,7,9
CMA					—	3		3,5,7,9	5,7,9
GNG						—		5,7,9	5,7,9
MBG							—	5,7,9	5,7,9
NSII								—	
SPE2									—

Figure 7.4: Results when solving the WFG7 problem. See Fig 7.1 for a description of each subfigure and abbreviations.



(a) $M = 3$ (b) $M = 5$ (c) $M = 7$ (d) $M = 9$

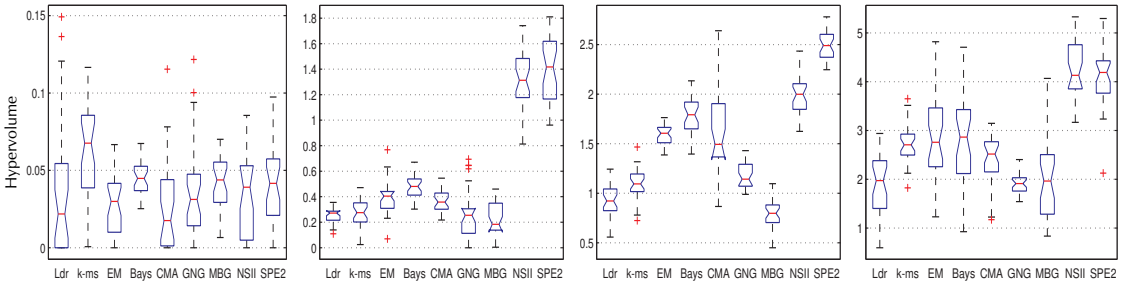


(e) Model–building CPU ops. (f) Objective function evals.

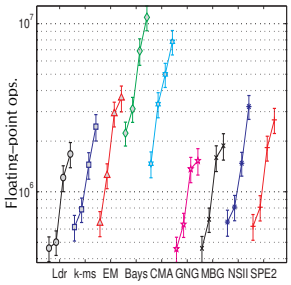
(g) Instances with statistically significant better results.

	Ldr	k-ms	EM	Bays	CMA	GNG	MBG	NSII	SPE2
Ldr	—	9	7,9	5,7,9	7			5,7,9	5,7,9
k-ms		—	7	5,7	7	7		5,7,9	5,7,9
EM			—					5,7,9	5,7,9
Bays				—				5,7,9	5,7,9
CMA					—			5,7,9	5,7,9
GNG						—		5,7,9	5,7,9
MBG							—	5,7,9	5,7,9
NSII								—	3,7
SPE2									—

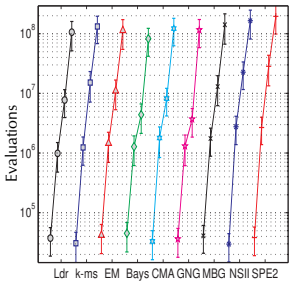
Figure 7.5: Results when solving the WFG8 problem. See Fig 7.1 for a description of each subfigure and abbreviations.



(a) $M = 3$ (b) $M = 5$ (c) $M = 7$ (d) $M = 9$



(e) Model–building CPU ops.



(f) Objective function evals.

(g) Instances with statistically significant better results.

	Ldr	k-ms	EM	Bays	CMA	GNG	MBG	NSII	SPE2
Ldr	—	3,7,9	5,7,9	3,5,7,9	5,7,9	7		5,7,9	5,7,9
k-ms		—	5,7	5,7	5,7			5,7,9	5,7,9
EM			—	3,5,7			3	5,7,9	3,5,7,9
Bays				—				5,7,9	5,7,9
CMA					—		3	5,7,9	3,5,7,9
GNG						—		5,7,9	5,7,9
MBG							—	5,7,9	5,7,9
NSII								—	7
SPE2									—

Figure 7.6: Results when solving the WFG9 problem. See Fig 7.1 for a description of each subfigure and abbreviations.

$\Phi_{p,m}$ is then computed as

$$P_{p,m}(A_i) = \sum_{j=1; j \neq i}^{K} \delta\left(A_i, A_j, \Phi_{p,m}\right). \tag{7.4}$$

This index should summarize the performance of each algorithm with regard to its peers.

Figure 7.7 exhibits the results computing the performance indexes. Figure 7.7a represents the mean performance indexes yielded by each algorithm when solving each problem in all of its configured objective dimensions,

$$\overline{P}_p(A_i) = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} P_{p,m}(A_i). \tag{7.5}$$

We have not included NSGA-II and SPEA2 in the plots as they were clearly outperformed by the other algorithms, and would, therefore, not be useful for presenting results. Nevertheless, their results were used to compute the performance indexes.

It is worth noticing that GNG and MB-GNG have better overall results than the other algorithms. It is somewhat unexpected that the randomized leader and the k-means algorithms do not have a very good overall performance for some problems, like WFG5 and WFG7 for the randomized leader and WFG8 and WFG9 for k-means. A possible hypothesis is that these results may be biased by the three-objective problems, where there are sizable differences compared with the results of the other dimensions.

This situation is clarified in Figure 7.7b, which presents the mean values of the index computed for each dimension

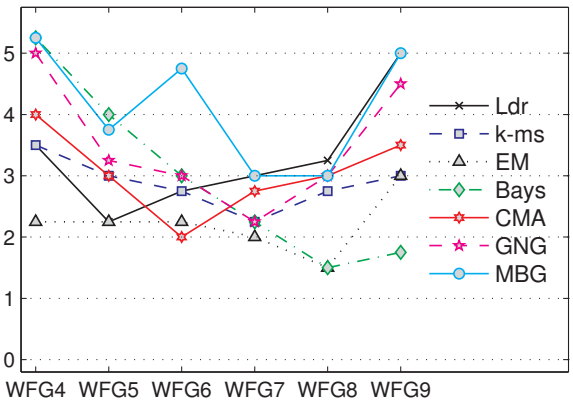
$$\overline{P}_m\left(A_i\right) = \frac{1}{P}\sum_{p=1}^{P} P_{p,m}\left(A_i\right) . \tag{7.6}$$

There is evidence that there is no substantial difference between the results yielded by the different algorithms in the three-objective case, as their index values are more uniform. It is also noticeable that CMA-ES seems to outperform all the other algorithms for all problems in this dimension. This panorama changes when inspecting the results in higher dimensionality (in the objective function space). In those cases the least statistically robust algorithms tend to perform comparatively better, with the exception of Bayesian networks that seem to improve as the number of dimensions increases, but, of course, at the expense of a great computational cost.

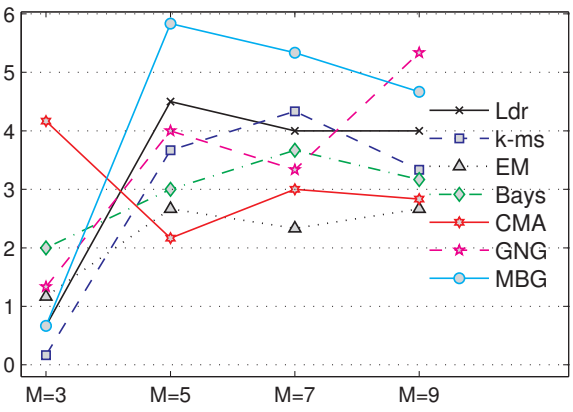
It is worthwhile analyzing the performance of MB-GNG. In most cases, MB-GNG outperformed the other algorithms in higher dimensionality. This outcome can be attributed to the fact that MB-GNG is the only algorithm that has so far been devised especially for the model-building problem.

7.4 Remarks on the Experiments

The experiments illustrated empirically that algorithms that have no statistical groundwork outperformed others that do. According to the hypothesis put forward in this theses, such behavior is caused by the fact that model-building has not yet been recognized as different from typical machine learning problems and, as such, having specific requirements that need to be met. The main aim of this work is to trigger further studies on this topic and, ultimately, new model-building algorithms.



(a) $\overline{P}_p\,()$, mean performance by problem.



(b) $\overline{P}_m\,()$, mean performance by number of objectives.

Figure 7.7: Mean values of the performance index calculated by problem and number of objectives across the different problems.

8

Assessing MONEDA

> I have been trained not to believe anything unless it can be demonstrated in the laboratory on rats or sophomores.
>
> *Steven Pinker*

An essential part of this work is to understand how MONEDA performs in practical situations and its outcome with similar state-of-the-art algorithms. MONEDA embeds the hypothesis related to the model-building issue presented in the previous theoretical discussion. Therefore, the analysis of the experimental results is indispensable for grasping a better understanding of the issue. That is why we now focus on solving a set of well-known problems with a selected set of the previously discussed evolutionary multi-objective optimizers, in particular, naïve MIDEA, MrBOA, RM-MEDA, MOPED, NSGA-II, SPEA2 and, of course, MONEDA.

We will now describe the experimental setup of our study in detail. First, we discuss the test problems used and the performance indicators applied. We then depict the hardware and software configurations and the choice of initial parameters of the applied algorithms.

8.1 Test Problems

Most experiments involving MOPs deal with only two or three objectives problems. In these experiments we intend to deal with higher-dimensional problems since we are interested in assessing the scalable behavior of MONEDA. Consequently, we have chosen for our analysis some of the members of the DTLZ family of scalable problems (Deb et al., 2004), in particular, the DTLZ3, DTLZ6 and DTLZ7 problems, and some of the WFG set of scalable problems (Huband et al., 2005), in particular WFG1, WFG2 and WFG6.

The DTLZ problems were selected for our experiments because of the relative simplicity of their specification and the existence of an a priori known Pareto-optimal front.

To complement the relative simplicity of the aforementioned problems we also dealt with a selection of the more complex walking fish group (WFG) problem set. The WFG problem set was devised as a particular instantiation. It consists of nine problems. From those problems, we chose WFG1, WFG2 and WFG6 because of their properties and inherent

complexities. Although these problems share the same formulation of their Pareto-optimal sets, the corresponding Pareto-optimal fronts each have a different appearance. WFG1 is a separable, uni-modal problem with polynomial and flat bias. Its Pareto-optimal front has a mixed convex geometry. WFG2 differs from WFG1 as it is non-separable, multi-modal and has a disconnected Pareto-optimal front. Finally, WFG6 is non-separable, uni-modal and has a concave Pareto-optimal front.

In all cases, the problems were configured in a progressive complexity fashion, increasing the number of objective space dimensions, specifically, 3, 6, 9 and 12 objective functions.

It should be noted that these are not the only problems that could be used for the type of experiments we carried out (see Huband et al. (2006a) and Coello Coello et al. (2007) for comprehensive reviews). Unfortunately, because of the high computational demands of such experiments and the limited amount of resources available we have been forced to restrain the study to fewer problems in order to be able to deal with a larger amount of objectives. We refer the interested reader to Appendix C where the problems here commented are described in full detail.

8.2 Experiment Design

Experiments were carried out under the PISA framework (Bleuler et al., 2003). The algorithms' implementations were adapted from the ones provided by their respective authors with the exception of NSGA-II and SPEA2, that were already distributed as part of the framework, and MOPED and MONEDA that were implemented from scratch. In all cases, the code was reviewed to ensure its optimality in order to have valid temporal complexity measurements.

A correct selection of each algorithm's initial parameter has a direct impact on the validity of experiments like the ones we are proposing. Normally, initial parameters are selected after an initial hand tuning with the small-scale problems. In this study, because of the high computational demands of the experiments, this adjustment phase has been limited to the three dimensional problems. Some parameters, however, needed to be bound to the dimension of the objective space, M. In such cases an explicit bounding formula was used. The parameters selected for each algorithm are summarized in Table 8.1. Whenever possible, we have used the parameters of algorithms as reported in their corresponding papers in order to ensure the reproducibility and comparison with previously published results.

Experiments were carried out in a 3.4 GHz Intel Quad-core computer with 4 GB of RAM memory running the Linux operating system. Each execution was repeated 30 times in order to have statistically significant results.

Stopping an optimizer is in itself a complex matter. It is usual practice in the evolutionary field to stop experiments after a given number of iterations. This strategy is of no use for this study since it is impossible to predict how much computation is required to solve high-dimensional problems. Because of that fact why we applied the MGBM multi-objective optimization stopping criterion described and tested on Chapters 5 and 6, respectively.

The quality of solutions in terms of approximation to the Pareto-optimal front is determined by using the hypervolume and additive epsilon indicator. These performance indicators require having a reference set of points. The the additive epsilon indicator use the

Table 8.1: Parameters of the algorithms used in the experiments.

Common parameters	
Population size (n_{pop})	$250 \cdot 10^{\frac{M}{3}-1}$
MONEDA	
Number of initial GNG nodes (N_0)	2
Maximum edge age ($\nu_{\max}$)	40
Best node learning rate (ϵ_{b})	0.1
Neighbor nodes learning rate (ϵ_{v})	0.05
Insertion error decrement rate (δ_{I})	0.1
General error decrement rate (δ_{G})	0.1
Accumulated error threshold (ρ)	0.2
Selection percentile (α)	0.3
$\hat{P}_t$ to $N_{\max}$ ratio (γ)	0.5
Substitution percentile (ω)	0.25
RM-MEDA	
Selection portion	0.3
Number of LPCA clusters	$\frac{10}{3}M$
Maximum training steps in LPCA	$\frac{20}{3}M$
Extension rate	0.25
MOPED	
Selection portion	0.3
Sampling parameter (τ)	2
Fitness parameter (α)	0.2

Naïve MIDEA	
Selection percentile (τ)	0.3
Diversity percentile (δ)	15
Number of parents of a variable (κ)	2
Maximum number of clusters	$\lceil 0.5 \lfloor \tau n_{\text{pop}} \rfloor \rceil$
Threshold for the leader algorithm	0.1
MrBOA	
Selection portion (τ)	0.3
Number of parents or a variable	5
Number of mixture components	3
Threshold of leader algorithm	0.3
NSGA-II	
Crossover probability (p_c)	0.7
Distribution index for SBX (η_c)	15
Mutation probability (p_m)	$\frac{1}{n_{\text{pop}}}$
Dist. index for polynomial mut. (η_m)	20
SPEA2	
Crossover probability (p_c)	0.7
Distribution index for SBX (η_c)	15
Mutation probability (p_m)	$\frac{1}{n_{\text{pop}}}$
Dist. index for polynomial mut. (η_m)	20
Ratio of pop. to archive sizes	4 : 1

Pareto-optimal front, $\mathcal{O}^*$, to carry out their calculations. Similarly, the hypervolume indicator needs a set of nadir points. These facts pose a problem when carrying out experiments that deal with many dimensions. This problem is particularly acute in the cases where the Pareto-optimal front is needed. In such cases, even when an explicit formulation of the front exists, it is computationally unviable to sample sufficiently well to obtain $\tilde{\mathcal{O}}^*$.

The test problems employed in this chapter, with the exception of DTLZ3 and WFG6, suffer from this inconvenience. In the case of DTLZ3 and WFG6, as their $\mathcal{O}^*$ lie on the first orthant of an hypersphere of radius 1 situated on the coordinates origin, it is straightforward to determine the distance from any point in the objective space to $\mathcal{O}^*$. However, for calculating the Pareto-optimal front coverage this feature is of no use as a sampled version of $\mathcal{O}^*$ is required.

To address this issue we have taken an alternative approach similar to the one used by the purity performance indicator (Bandyopadhyay et al., 2004; Ishibuchi and Murata, 1998). A combined set $\mathcal{PF}^+$ is defined as the union of the solutions obtained from the different algorithms across all the experiment executions. $\tilde{\mathcal{O}}^*$ is then determined by extracting the non-dominated elements

$$\boldsymbol{x} \in \tilde{\mathcal{O}}^* \text{ iff } \boldsymbol{x} \in \mathcal{PF}^+ \text{ and } \nexists \boldsymbol{y} \in \mathcal{PF}^+ \text{ such that } \boldsymbol{y} \prec \boldsymbol{x}. \tag{8.1}$$

Although this procedure circumvents the problems of performing a direct sampling of the Pareto-optimal front shape function, special precautions should be taken when interpreting the results. It should be noticed that the algorithm performance will be measured with regard to the set of overall best solutions; not against the actual Pareto-optimal front.

A similar method is used for determining the nadir points set used by the hypervolume indicator. In this case $\mathcal{N}$ is computed as

$$\boldsymbol{x} \in \mathcal{N} \text{ iff } \boldsymbol{x} \in \mathcal{PF}^{+} \text{ and } \nexists \boldsymbol{y} \in \mathcal{PF}^{+} \text{ such that } \boldsymbol{x} \prec \boldsymbol{y}\,. \tag{8.2}$$

The statistical validity of the results is assessed using the Connover–Inman procedure that is described in Appendix B.

8.3 Quality and Performance Analysis

The statistical description of the final results yielded by the runs of algorithms when dealing with the DTLZ problems can be inspected on figures 8.1, 8.2 and 8.3 in the form of box plots (Benjamini, 1988; Chambers et al., 1983) and as tables that summarize the outcome of the statistical tests described before. This representation assists in the assessment of the quality and validity of the final solutions of the algorithms. In all cases the performance was computing using the results obtained at the iterations marked by the stopping criterion. Appendix D shows the progression of the mean of each indicator value as iterations advanced.

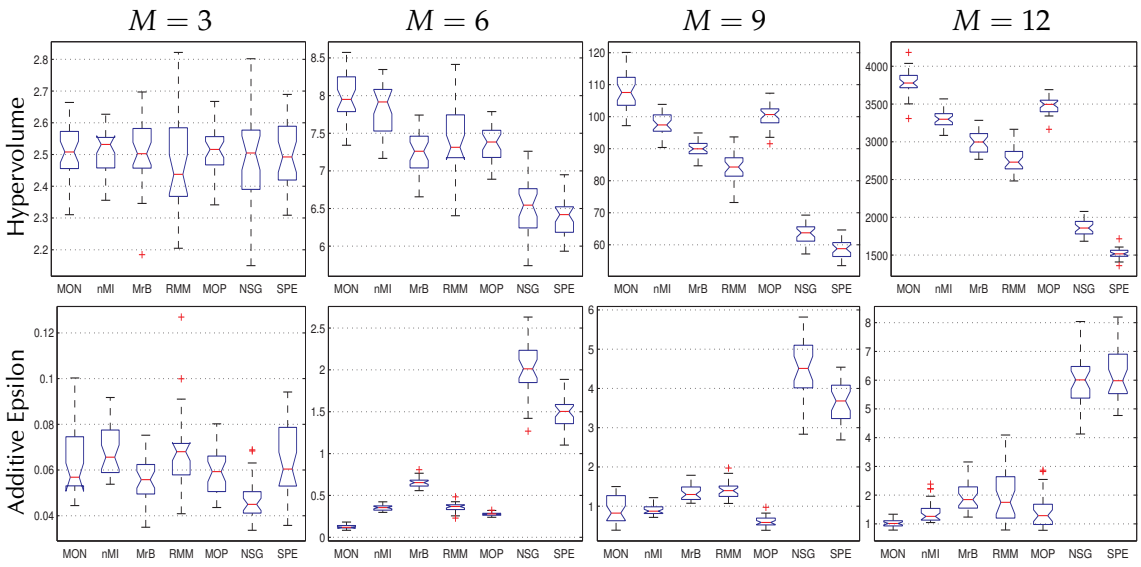
In the three-dimensional configurations, MONEDA performed similarly to the rest of the algorithms. This was an expected outcome since our MOEDA uses an already existent fitness function and its model-building algorithm is meant to provide a significant advantage in more extreme situations.

However, as the tests proceed into higher dimensions, it becomes evident that MONEDA outperforms the rest of the applied optimizers. MONEDA not only yields better final results but they have a very low variance and also less iterations are required. This means that it performed consistently well across the different runs. The results not only show that MONEDA's solutions are close to the optimal but also they manage to evenly cover the Pareto-optimal front.

In spite of the encouraging results described here, special care should be taken when analyzing them. It should be noticed that because of the methodology used for generating the surrogate Pareto-optimal front, $\tilde{\mathcal{O}}^{*}$, the ϵ-indicator is not contrasting the solutions against the true optimal front. This could potentially lead to improper conclusions.

Under this light, the outstanding results of MONEDA should be interpreted in a relative fashion. MONEDA's low $I_{\epsilon+}$ values indicate that its solutions have been better than most of the solutions obtained from the other algorithms and therefore they belong to the joint local Pareto-optimal front, $\tilde{\mathcal{O}}^{*}$. Similarly, although MONEDA has managed to produce better results when compared to the rest of the algorithms, it is not possible to assert that its solutions are sufficiently close to the Pareto-optimal front. The scheme applied for determining $\tilde{\mathcal{O}}^{*}$ may also be the cause of the particularly low variance of the MONEDA results.

Nevertheless, when analyzing the variances of the indicators, a most interesting side discussion is prompted. On one hand, the $I_{\epsilon+}$ indicator report relatively low variances for most algorithms, been particularly small for MONEDA. On the other, the measurements of I_{H} are more spread. Again, this is probably caused by the scheme we used for determining $\tilde{\mathcal{O}}^{*}$. From this situation an initial conclusion can be deduced: for experiments like the ones we are analyzing here it is of more use to apply performance indicator that do not depend on



(a) Graphical representation of the indicators as box plot.

(b) Results of the statistical test for the hypervolume indicator.

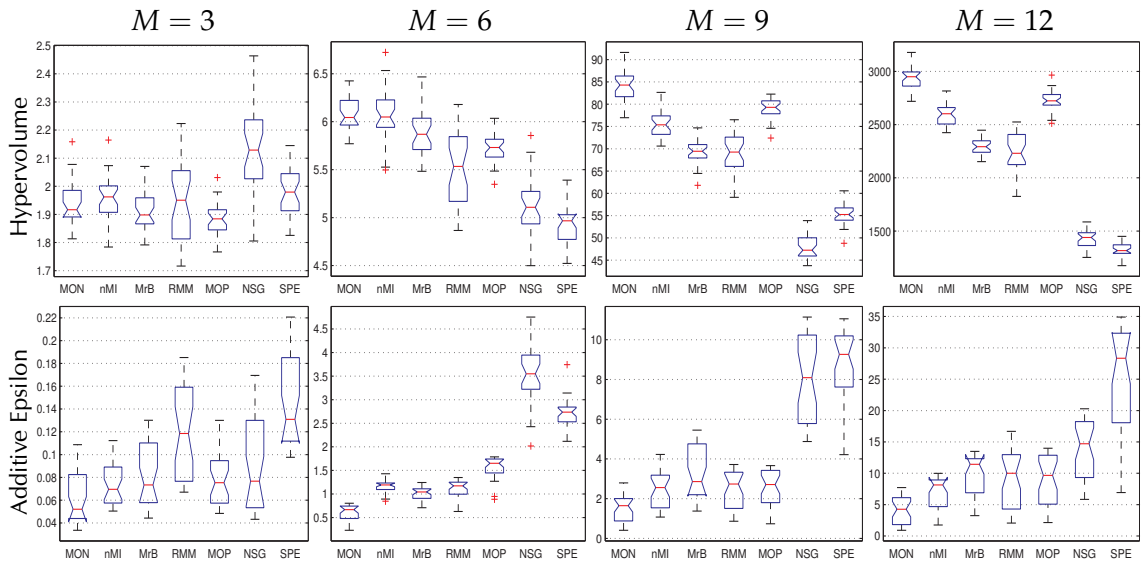
	MON	nMI	MrB	RMM	MOP	NSG	SPE
MONEDA	—	9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12
naïve MIDEA		—	6, 9, 12	6, 9, 12	6	6, 9, 12	6, 9, 12
MrBOA			—	9, 12		6, 9, 12	6, 9, 12
RM–MEDA				—		6, 9, 12	6, 9, 12
MOPED					—	6, 9, 12	6, 9, 12
NSGA–II						—	9, 12
SPEA2							—

Figure 8.1: Summary of the statistical description of the results yielded by the MONEDA (MON), naïve MIDEA (nMI), MrBOA (MrB), RM-MEDA (RMM), MOPED (MOP), NSGA-II (NSG) and SPEA2 (SPE) algorithms when solving the DTLZ3 problem. Figure 8.1a presents the indicator values obtained after each experiment as box plots. Table 8.1b summarizes the outcome of performing the statistical hypothesis tests. The numbers shown are the problem dimension where the test detected a statistically significant better indicator values of the algorithm in each row with respect of those in the columns.

a known Pareto-optimal front, as the I_{H} indicator. Anyhow, in spite of the higher variances, MONEDA still yields better and more statistically sound results than the rest of the algorithms.

In spite of the relevant information that can be extracted from the above-discussed visual representation, a more formal approach is necessary to confirm from a statistical point of view the previous results. The results of applying the Connover–Inman procedure (see Appendix B) to the outcome of the previous experiments is summarized on Tables 8.1b, 8.2b and 8.3b for the DTLZ3, DTLZ6 and DTLZ7 problems, respectively. In those tables it can be verified that in most experiments of 6 objectives and beyond MONEDA was able to tackle the problem better than rest of the algorithms.

Similar conclusions can be drawn from the experiments involving the WFG1, WFG2 and WFG6 problems. Although, in general terms, these problems pose a bigger challenge to the optimizers the progress shapes of the algorithms are rather similar to those of the previous



(a) Graphical representation of the indicators as box plots.

(b) Results of the statistical test for the hypervolume indicator.

	MON	nMI	MrB	RMM	MOP	NSG	SPE
MONEDA	—	9, 12	6, 9, 12	6, 9, 12	3, 6, 9, 12	6, 9, 12	6, 9, 12
naïve MIDEA		—	3, 6, 9, 12	6, 9, 12	3, 6	6, 9, 12	6, 9, 12
MrBOA			—	6	6	6, 9, 12	6, 9, 12
RM–MEDA				—		6, 9, 12	6, 9, 12
MOPED					—	6, 9, 12	6, 9, 12
NSGA–II						—	3, 6, 12
SPEA2							—

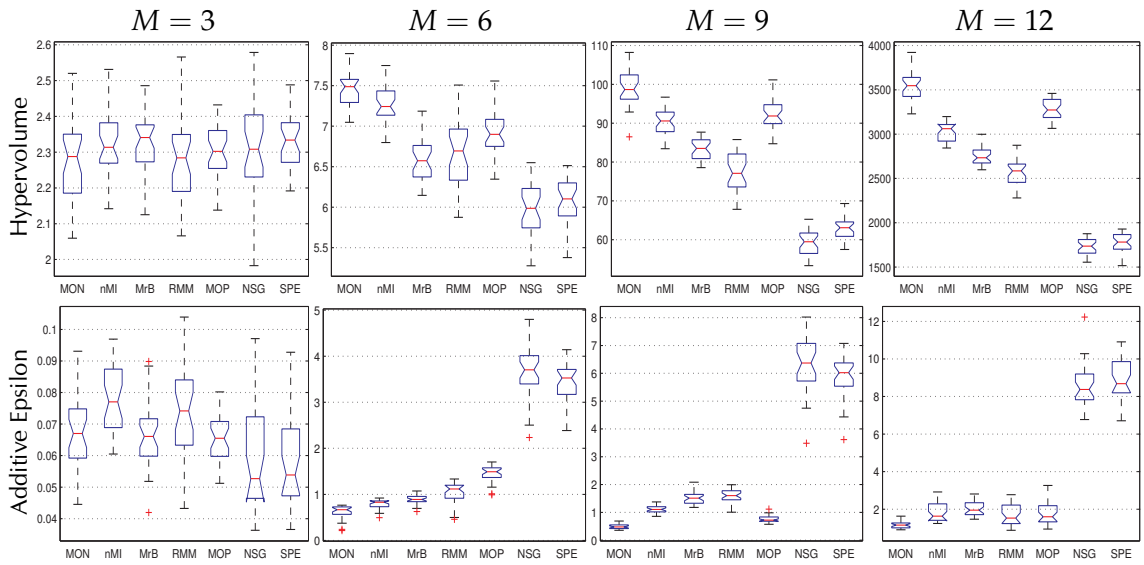
Figure 8.2: Statistical description of the results yielded by the algorithms involved in the experiments when solving the DTLZ6 problem. See Figure 8.1 for an extended description.

problems. This fact can be appreciated on Figures 8.4, 8.5 and 8.6 respectively, which depicts the statistical properties of the indicator values yielded by each algorithm.

The results for the WFG problems share the same properties exhibited by previous ones. Even though the scalar values of the indicators change, the outcome of comparing the algorithms performances is more or less the same.

The critical assessment of these results lead us to hypothesize that thanks to its novel treatment of the outliers in the model-building data set our approach manages to overcome the difficulties that hampers the rest of the methods. Although very interesting, the results presented here raise the question: to at what degree are they conditioned by the particularities of the solved problems? This issue must be investigated further, order to understand if the low dispersion of the error indicators can only be obtained in the solved problems, or if it can be extrapolated to other more complex problems as well.

In any case, one of the most important conclusions that can be drawn from these experiments is that MONEDA has shown itself to be a robust algorithm. MONEDA, in spite of having a relatively large number of parameters, is capable of dealing with a wide range of problems, each one with different characteristics, without having to resort to a custom tuning



(a) Graphical representation of the indicators as box plots.

(b) Results of the statistical test for the hypervolume indicator.

	MON	nMI	MrB	RMM	MOP	NSG	SPE
MONEDA	—	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12
naïve MIDEA		—	6, 9, 12	6, 9, 12	6	6, 9, 12	6, 9, 12
MrBOA			—	3, 9, 12		6, 9, 12	6, 9, 12
RM–MEDA				—		6, 9, 12	6, 9, 12
MOPED					—	6, 9, 12	6, 9, 12
NSGA–II						—	
SPEA2							—

Figure 8.3: Statistical description of the results yielded by the algorithms involved in the experiments when solving the DTLZ7 problem. See Figure 8.1 for an extended description.

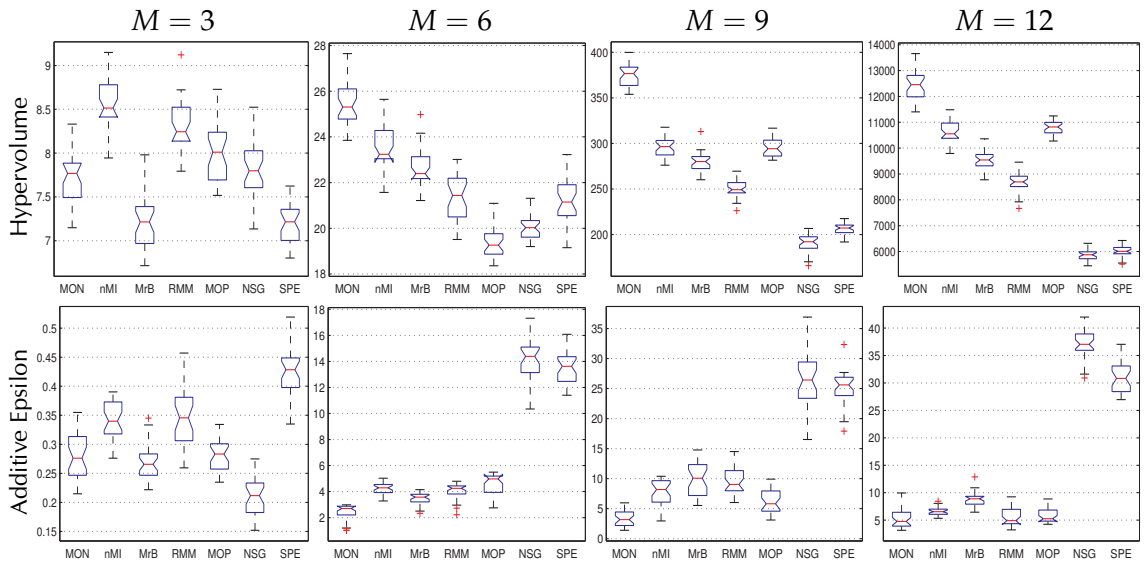
of it configuration.

Asserting these facts is rather difficult as it implies cross-examining and comparing the results presented separately. This is a similar situation to that reported in the experiments of Chapter 7. That is why we also applied here the integrative representation used in Section 7.3.

Figure 8.7 exhibits the results computing the performance indexes. Figure 8.7a represents the mean performance indexes yielded by each algorithm when solving each problem in all of its configured objective set dimensions.

It is worth noticing that MONEDA have better overall results with respect to the other algorithms. It is somewhat unexpected that the randomized leader and the k-means algorithms do not have a very good overall performance for some problems, like WFG5 and WFG7 for the first one and WFG8 and WFG9 for the second. It can be hypothesized that these results can be biased by the three objective problems, having dramatic differences in their results with respect to the rest of the dimensions considered.

This situation is clarified in Figure 8.7b, which presents the mean values of the index



(a) Graphical representation of the indicators as box plots.

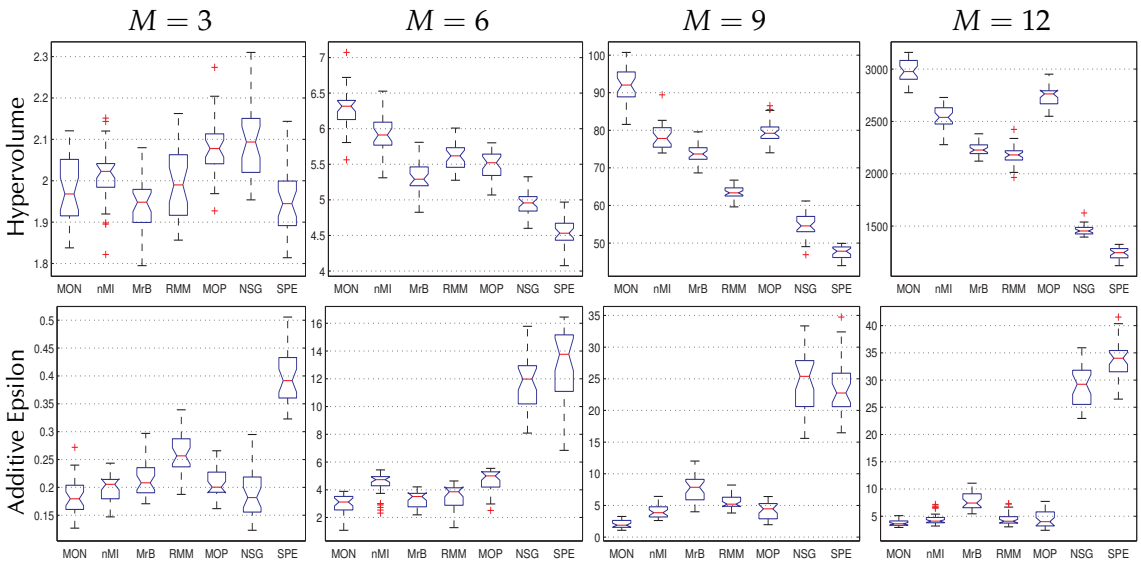
(b) Results of the statistical test for the hypervolume indicator.

	MON	nMI	MrB	RMM	MOP	NSG	SPE
MONEDA	—	6, 9, 12	3, 6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	3, 6, 9, 12
naïve MIDEA		—	3, 6, 9, 12	3, 6, 9, 12	3, 6	3, 6, 9, 12	3, 6, 9, 12
MrBOA			—	6, 9, 12	6	6, 9, 12	6, 9, 12
RM–MEDA				—	3, 6	3, 6, 9, 12	3, 9, 12
MOPED					—	9, 12	3, 9, 12
NSGA–II						—	3
SPEA2							—

Figure 8.4: Statistical description of the results yielded by the algorithms involved in the experiments when solving the WFG1 problem. See Figure 8.1 for an extended description.

computed for each dimension, $\overline{P}_m$. In this case, it can be corroborated that in the three objective case the is no substantial difference in the results produced by the different algorithms, as they have more evenly shared values in their indexes. It is also noticeable that CMA-ES seems to outperform all the other algorithms in all problems in this dimension. This panorama changes when inspecting the results in higher dimensionality (in objective function space). In those cases the least statistically robust algorithms tend to perform comparatively better with the exception of Bayesian networks that seem to improve as the number of dimensions increases, but, of course, at the expense of a great computational cost.

It is relevant to analyze the performance of MB-GNG. In most cases, MB-GNG outperformed the rest of the algorithms in higher dimensionality. This corroborates the results presented by us in previous works (Martí et al., 2008a, 2009d). This outcome can be attributed to the fact that MB-GNG is the only algorithm that has been devised so far, having the model-building problem in mind.



(a) Graphical representation of the indicators as box plots.

(b) Results of the statistical test for the hypervolume indicator.

	MON	nMI	MrB	RMM	MOP	NSG	SPE
MONEDA	—	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12
naïve MIDEA		—	3, 6, 9, 12	6, 9, 12	6	6, 9, 12	3, 6, 9, 12
MrBOA			—	9, 12		6, 9, 12	6, 9, 12
RM–MEDA				—		6, 9, 12	6, 9, 12
MOPED					—	6, 9, 12	3, 6, 9, 12
NSGA–II						—	3, 6, 9, 12
SPEA2							—

Figure 8.5: Statistical description of the results yielded by the algorithms involved in the experiments when solving the WFG2 problem. See Figure 8.1 for an extended description.

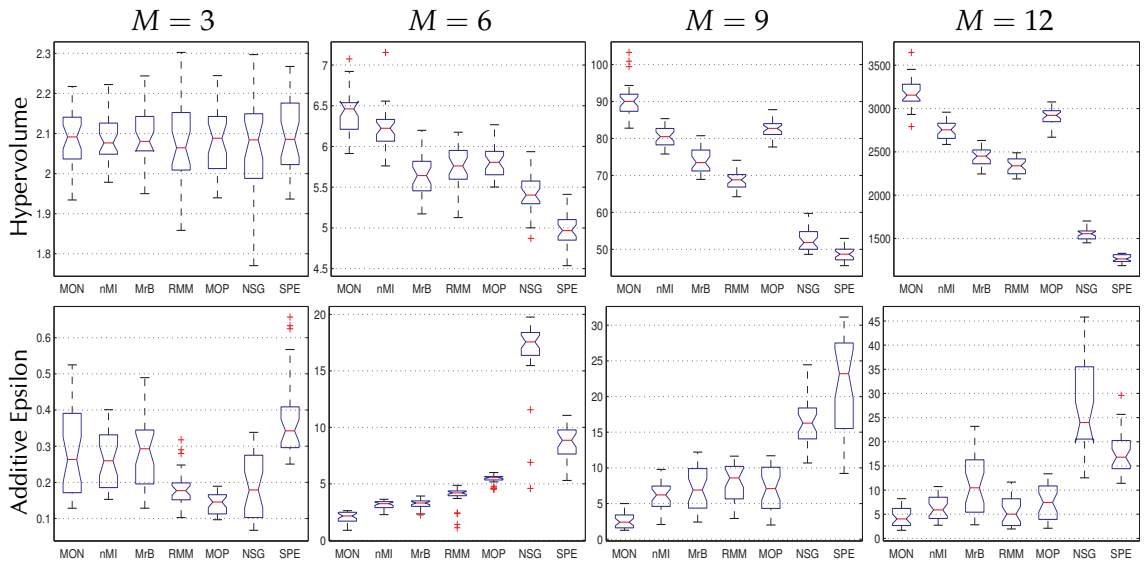
8.4 Computational Cost of the Algorithms

On of our main concerns when comparing the different algorithms was their computational requirements. One simple and illustrative way of doing this is to plot the progress of the different algorithms when solving each problem. A discussion on how to carry this out is presented in Section B.1.

Figure 8.8 summarizes the average amount of iterations used by each algorithm, the mean floating-point CPU operations per iteration and the mean total floating-point CPU operations used by the algorithms when solving the DTLZ problems. Figure 8.9 contains the homologous analysis for the WFG problems.

This set of measurements reinforces the conclusions obtained so far. It can be inferred from the figures that, besides requiring relatively few iterations to converge, it also expends fewer operations on every iteration.

The cases of MrBOA and RM-MEDA are very illustrative. Although they require fewer iterations, their mean CPU operations per iteration are the highest and, correspondingly, their total amount of CPU operations is in the same situation. This makes the case for not



(a) Graphical representation of the indicators as box plots.

(b) Results of the statistical test for the hypervolume indicator.

	MON	nMI	MrB	RMM	MOP	NSG	SPE
MONEDA	—	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12	6, 9, 12
naïve MIDEA		—	6, 9, 12	6, 9, 12	6	6, 9, 12	6, 9, 12
MrBOA			—	9, 12		6, 9, 12	6, 9, 12
RM–MEDA				—		6, 9, 12	6, 9, 12
MOPED					—	6, 9, 12	6, 9, 12
NSGA–II						—	6, 9, 12
SPEA2							—

Figure 8.6: Statistical description of the results yielded by the algorithms involved in the experiments when solving the WFG6 problem. See Figure 8.1 for an extended description.

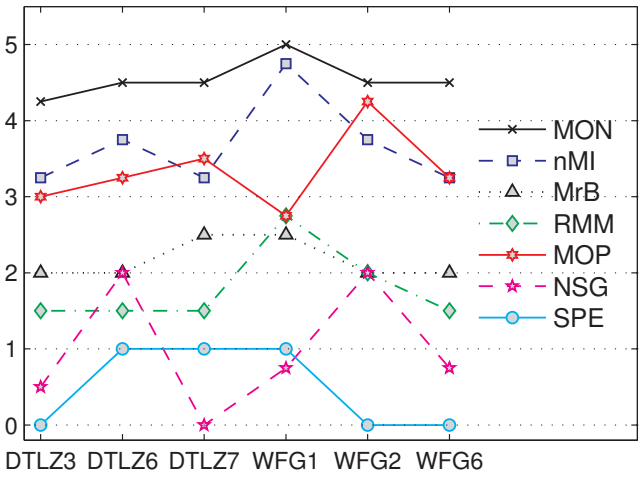
measuring only the amount of iterations or the number of function evaluations in this class of experiments.

Another interesting phenomenon is the relatively low increase in the number of iterations when moving from 6 to 9 objectives. This behavior is shared across most algorithms and problems. In our opinion it can be attributed to the relatively large size of the population used.

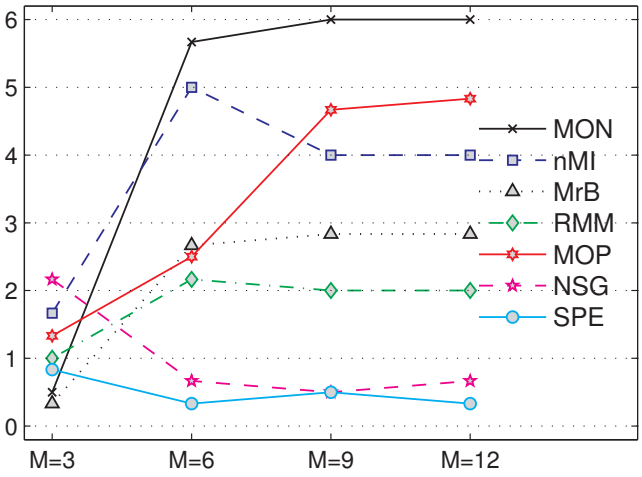
Also noticeable, is the (presumably) exponential increase on the amount of iterations and CPU consumption as the problem complexity grows. This means that future algorithms should be aware of this problem and at least try to alleviate this growth.

8.5 Commentaries about the results

It can be argued that, as in any experimental comparison, the parameters of the different algorithms have strongly biased the results. For example, it could be hypothesized that NSGA-II and SPEA2, with much larger population sizes, would probably yield better approximations



(a) $\overline{P}_p\,(\,)$, mean performance by problem.



(b) $\overline{P}_m\,(\,)$, mean performance by number of objectives.

Figure 8.7: Mean values of the performance index calculated by problem and number of objectives across the different problems.

to the Pareto-optimal fronts, albeit at a higher computational expense.

In any case, the most remarkable conclusion, when assembling the results exposed in Sections 8.3 and 8.4, is that MONEDA is capable of consistently producing similar or better results with regard to similar approaches at a lower computational cost. On its turn, this improvement can only be attributed to the introduction of a novel model-building algorithms specially made for the task, since other algorithm properties, like, for example, the fitness assignment, have been kept.

To test the exposed arguments we introduced a novel evolutionary algorithm called multi-objective optimization neural estimation of distribution algorithm (MONEDA). MONEDA puts forward an innovative neural network-based scheme for model-building. In particular, a modified GNG neural network is applied. This model-building algorithm addresses two theoretical and practical issues not taken into account by previous approaches.

MONEDA's behavior has been assessed by dealing with a set of well-known community-

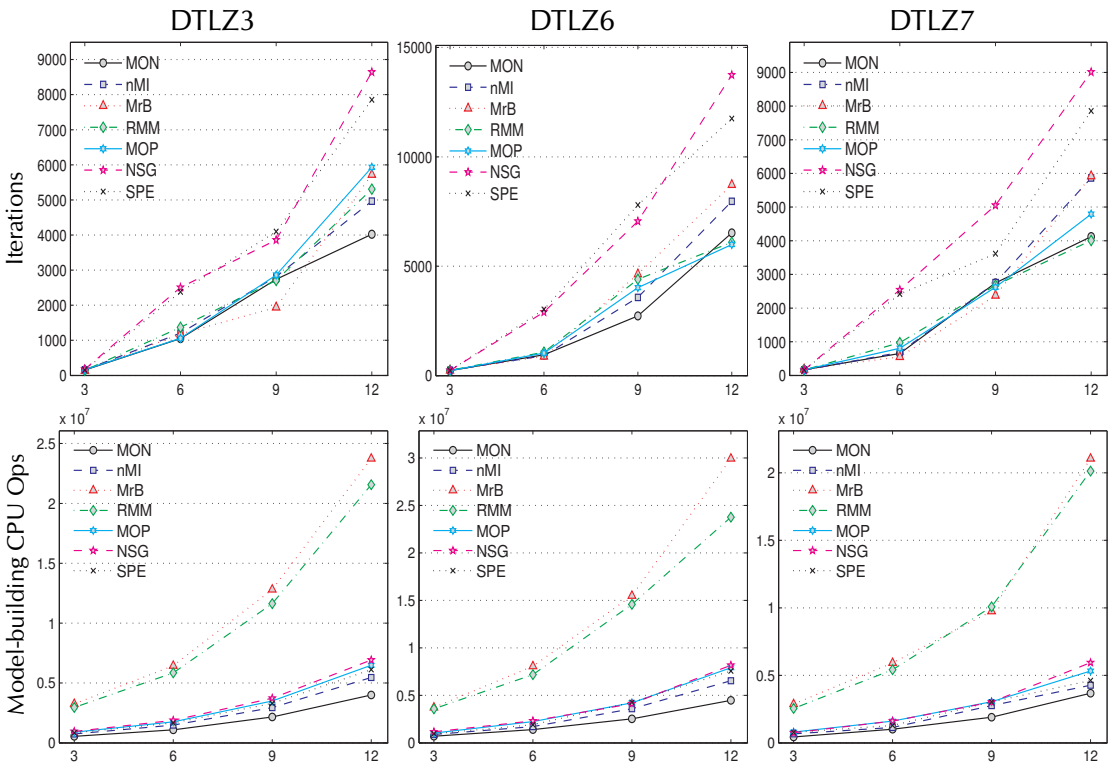


Figure 8.8: Comparative analysis of the computational cost of the algorithms under study when dealing with the DTLZ3, DTLZ6 and DTLZ7 problems. For each algorithm/problem/number of objectives combination the mean of the cost indicators yielded by each of the 30 runs is computed. The first row represents the mean number of iterations used by each algorithm, while the second, the mean of the intra-iteration floating-point CPU operations used for model-building.

accepted problems with a progressive increase in the number of objectives. Its results have been compared against a range of state-of-the-art algorithms.

The experimental results show that in problems with relatively few dimensions MONEDA performs similarly to other approaches. However, as the problem complexity scales, MONEDA outperforms the rest of the algorithms in terms of the quality of the solutions and their computational complexity.

However, there are many issues that remain open. For example, the algorithm's sensitivity to its parameters must be explored in depth. One of the main drawbacks of MONEDA is its rather large number of free parameters. Improvements should be made in order obtain a less parameterized algorithms.

It is equally important to grasp a more extensive range of test problems. These experiments, although very costly in terms of computational resources, are essential for understanding the reach of the improvements put forth here.

Similarly, there is plenty of room for improvement in MONEDA. Different combinations of selection and replacement schemes would probably produce better results. Similarly, there are different fitness assignment strategies, like the ones employed by SPEA2 or PAES, the ones based on relaxed forms of Pareto dominance and the ones based on performance indicators that should be contrasted in order to determine their suitability. Other interesting lines of

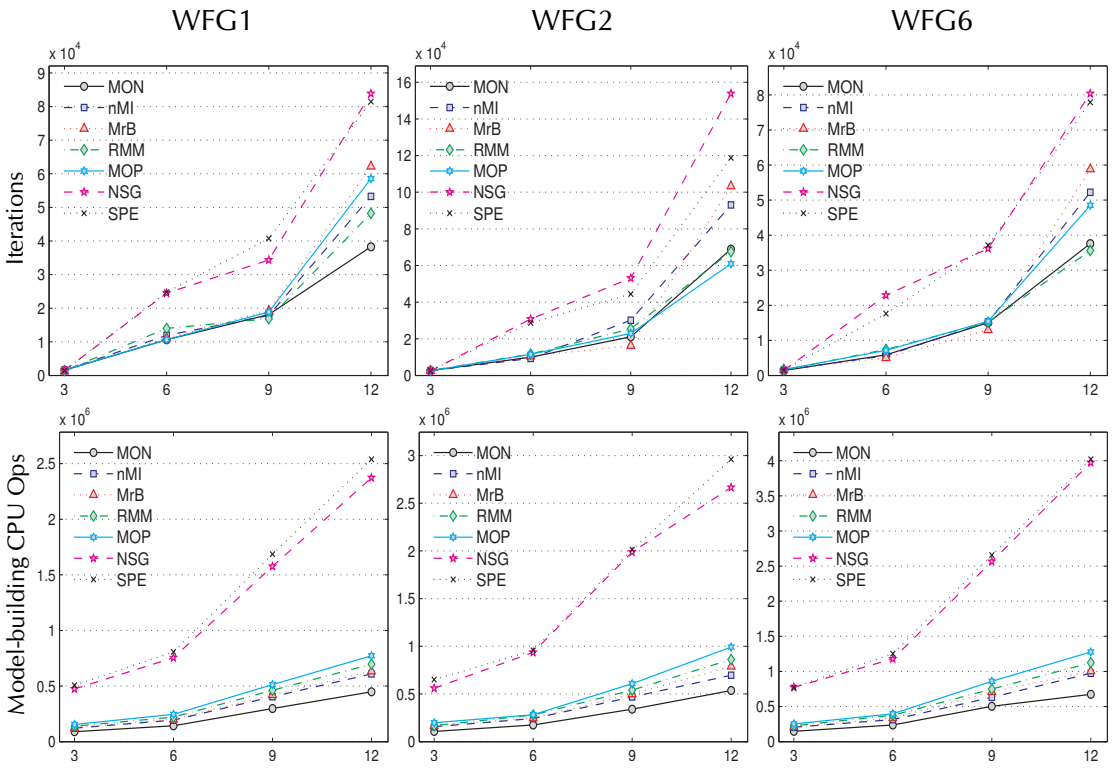


Figure 8.9: Comparative analysis of the computational complexity of the algorithms under study when dealing with the WFG1, WFG2 and WFG6 problems. For each algorithm/problem/number of objectives combination the mean of the cost indicators yielded by each of the 30 runs is computed. The first row represents the mean number of iterations used by each algorithm, while the second, the mean of the intra-iteration floating-point CPU operations used for model-building.

research are the reuse of models across iterations and the fusion of the fitness assignment and the model-building processes into a combined process that would be less computationally demanding. In this regard we have concentrated on the improvement model-building of the model-building algorithm, but still we acknowledge that further improvements can be obtained after introducing these alternatives to fitness assignment.

Beyond the successful outcome of experiments, the most important consequence of this work is that we have exposed a previously overlooked issue in the EDA field. To the best of our knowledge, the model-building analysis and the proposal we have presented have not been done before. Therefore, this work would be most valuable if it induces the emergence of a set of new approaches to the model-building issue. A more exhaustive review of suitable machine learning methods taking the considerations put forward into account would probably yield even better model builders. Perhaps even new methods should be synthesized in order to properly address this task.

Although we have focused on the multi-objective case, the discussed model-building issue can also be extended to single-objective EDAs. In cases where the optimizer must yield more than one optimal solution, such as in multi-modal optimization ones, the model-building issue should also manifest itself. In those cases, the considerations about the incorrect treatment of outliers also have a solid ground. These matters, however, open another

line of research, which is out of the scope of this chapter.

9

Conclusions

> The "Strange Loop" phenomenon occurs whenever, by moving upwards (or downwards) through levels of some hierarchical system, we unexpectedly find ourselves right back where we started.
>
> *Douglas Hofstadter — Gödel, Escher, Bach: An Eternal Golden Braid*

MULTI-OBJECTIVE optimization with evolutionary algorithms is one of the main topics of current research in the evolutionary area. The algorithms carrying out this task have prompted the development of new approaches and techniques, in part because of the challenging nature of the problem. Many-objective problems are of particular interest because of severe demands they impose on the algorithms solving them.

The leit motif of this thesis was to devise new algorithms that are suitable for solving those problems. The mainstream approaches to this issue focus on the development of better fitness assignment schemes or the reduction of objectives. Instead, we have decided to focus on how to provide better search engines for the optimization process.

In route to that purpose we have found that the problem of detecting when a MOEA could be stopped was not properly studied. This matter, although shared by applications involving all amount of objectives gain relevance when facing many-objective ones, since the usual strategy of just letting the algorithm run for a sufficiently large number of iterations is impracticable.

Regarding these issues, the main contributions of the thesis can be summarized as:

- *Understanding of the model–building issue*: we have hypothesized that current MOEDAs and, in particular, their model-building algorithms, are unsuitable for the problem in question. We have enumerated a number of features of current approaches that could be causing current approaches to underperform. In particular, we have proposed that the incorrect treatment of data outliers; the loss of population diversity; and the excess of computational effort devoted to finding an optimal population model might hampering the obtention of adequate results.

- *Proposal of a novel MOEDA algorithm*: thanks to the comprehension of the model-building issue we have proposed a new method that we expected that would deal

correctly with it. This new model-building growing neural gas showed in the experimental studies that it is able to outperform current state-of-the-art alternatives in high-complexity problems. When embedded in its corresponding MOEDA, the multi-objective neural EDA, the algorithm was able to yield similar or better results than similar approaches.

- *Introduction of a stopping criterion for multi-objective optimization methods*: the MGBM, at the time of its proposal, was the only stopping criterion applicable to all evolutionary approaches. In spite that, in subsequent years, other approaches have been proposed, MGBM has shown to consistently yield correct answers.

9.1 Towards a Paradigm Shift in EDAs

The results of this work prompt a series of considerations that we believe could be used as the basis for a possible paradigm shift within MOEDAs. One of the main conclusions is that model-building algorithms without a solid statistical foundation generally outperform the others for problems with a dimensionality greater than three (in the objective function space). These results, therefore, sustain the hypothesis put forward in Chapters 3 and 4. It is now more evident that the model-building problem has different characteristics to other existing machine learning problems.

The improvement achieved with the application of MB-GNG is particularly noteworthy. Although it could be argued that the custom-designed MB-GNG yields substantially better results with respect to current alternatives, we find that there is still a lot of room for improvement in this area. Therefore, a more fruitful debate would be around how to create algorithms that are capable of properly dealing with the model-building issue.

It is indeed true that the curse of dimensionality cannot be avoided in the long term. Similarly, the no-free-lunch theorem in the multi-objective case has shown that there will be no universal multi-objective optimizer that outperforms all the other algorithms in all cases (Corne and Knowles, 2003). However, if we analyze the issues debated in this thesis and in the light of the experimental results presented here, we can point out different directions that may be pursued in order to achieve a substantial improvement in the MOEDA area.

As stated previously, one of the main causes of the current limitations of MOEDAs can be attributed to their disregard of outliers. In turn, this behavior can be put down to the error-based learning approaches that take place in the underachieving MOEDAs.

Error-based learning is rather common in most machine learning algorithms. It implies that model topology and parameters are tuned in order to minimize a global error measured across the learning data set. This type of learning of isolated data is not taken into account because these data contribute little to the overall error and, therefore, do not take an active part in the learning process.

This behavior makes sense in the context of many problems, as isolated data can be interpreted as being spurious, noisy or invalid. However, as we argued in Chapter 3, this is not the case in model building. In model building, all data are equally important, and, furthermore, isolated data might have a greater significance as they represent unexplored regions of the current optimal search space. This assessment is supported by the fact that most

of the better-performing approaches do not follow the error-based scheme. For this reason, perhaps another class of learning, such as instance-based learning (IBL) (Kibler et al., 1989; Quinlan, 1993) or match-based learning (Grossberg, 1982) would yield a sizable advantage. As a matter of fact, the leader and k–means algorithms are good representatives of IBL.

Another strategy of interest is the fusion of the information present in both the decision and objective sets. Most MOEDAs construct their models by exploiting only the decision variable space information, since the resulting model can be used for sampling new individuals. To the best of our knowledge, the only MOEDA work that has addressed this issue is related to the use of the multi-objective hierarchical BOA (mhBOA) (Pelikan et al., 2005, 2006b). MhBOA performs a k-means clustering of the local Pareto front obtained after applying the NSGA-II ranking function. Then, a local model is built for each cluster. It is worth remarking that a simpler approach would be to replace the NSGA-II's ranking function for one based on SPEA2, which has an embedded clustering process. Nevertheless, the underlying idea here is that the model would benefit from taking into account the properties of the individuals in both spaces.

Model reuse across iterations is another important issue. The most popular approaches so far either (i) create and later discard new models in every iteration or (ii) infer some of the most costly properties (such as the network topology in Bayesian networks) beforehand and tune the others in each iteration.

The first solution has the obvious drawback of wasting resources when large parts of the model are likely to be able to be reused across iterations. On the other hand, the other approach does not take into account the evolution of the local Pareto-optimal front and set as the optimization process progresses. To get MOEDAs with better scalability, the model-building algorithms must be able to handle some degree of reusability and, therefore, minimize the amount of computation carried out in each iteration.

In any case, it is clear from the above discussions and experiments that the model-building problem warrants a different approach that takes into account the particularities of the problem being solved. The ultimate solution to this issue is, perhaps, to create custom-made algorithms that meet the specific requirements of the problem at hand.

9.2 Future Work

The matters discussed as part of this thesis are by no means near a conclusive state. Perhaps the most important outcome of this thesis is to motivate further studies on the areas of study. Furthermore, in our opinion, this work is more useful as a reference point for a further research.

There are some areas that seem to be particularly interesting for further investigation:

- Better theoretical understanding and support of the model-building issue.
- Investigate other learning paradigms that could handle properly that task.
- Gain a better understanding of the evolutionary processes in order to judge during the execution of an algorithm the possibility of reaching a satisfactory solution or if it will not.

In these regards, we have already started to obtain some results, in particular in the second and third directions.

9.2.1 Towards Novel Learning Paradigms in Model Building

We are currently working on introducing new learning paradigms. For example, in a recent series of works (Martí et al., 2010b,d) we have introduced an adaptive resonance theory neural network with success.

Adaptive resonance theory (ART) (Grossberg, 1982) is a theory of human cognition that has seen a realization as a family of neural networks. It relies on a learning scheme denominated match-based learning and on intrinsic topology self-organization. These features makes it interesting as a case study as model-building approach. Match-based learning equally weights isolated and clustered data (Sarle, 1995), and, therefore, the algorithm does not disregard outliers. Similarly, self-organization makes possible the on-the-fly determination the model complexity required to correctly represent the data set, thus eliminating the need of an external algorithm for that task.

In this regard, we argue that error-based learning, the class of learning most commonly used in MOEDAs, is responsible for current MOEDA underachievement. We discuss in detail ART-based learning as a viable alternative and present a novel algorithm called *multi-objective ART-based EDA* (MARTEDA) that uses a Gaussian ART neural network (Williamson, 1996) for model-building and an hypervolume-based selection as described for the hypervolume estimation algorithm for multi-objective optimization (HypE) (Bader and Zitzler, 2008). We experimentally show that thanks to MARTEDA's novel model-building approach and an indicator-based population ranking the algorithm it is able to outperform similar MOEDAs and MOEAs.

9.2.2 Understanding Evolutionary Processes: The Fitness Homogeneity Indicator

A recent approach to stopping criteria (Martí et al., 2010e) focuses on the properties of what is a "healthy" population. Therefore, when the properties of a population deviate from the adequate configuration, the optimization process should be aborted.

It has been shown by different authors (i.e. (Martí et al., 2007, 2009a; Trautmann et al., 2009; Wagner et al., 2009)) that multi-objective optimization stopping criteria must consist of two processes. A local one, that we will call progress indicator, that determines at the end of an iteration how the supervised algorithm is performing. The second one, called evidence gathering process, takes care of combining these local measurements in order to gain a global scope of their trend and decide when to stop. It should be noted the difference between the term progress indicator, as used in the context of this work, and the term performance indicator.

However, by analyzing their theoretical and practical studies it comes up that little attention has been dedicated to the problem of detecting failures in these evolutionary processes, or in other words, when an the execution of a MOEA will not lead to a valid solution.

The conditions that could lead to a failed execution can be inferred from previous studies. One obvious condition consists of the incorrect selection of the different building blocks of

the evolutionary algorithm, like mutation, selection and mating operators and their parameters, something that has been previously studied in depth.

Another condition, that, although can be viewed as a particular class of the previous, calls for further study, is the allocation of a correct population size. In a series of experimental studies, like those of Knowles and Corne (2007); Purshouse and Fleming (2007), among others, it has been shown that there is an exponential dependence between the dimension of the objective space and the amount of resources required to solve the problem correctly. Therefore, if such dependency is not met it can be inferred that the problem could not be correctly solved. However, the exact nature of this dependency is particular to each problem and algorithm and, therefore, the results of those studies are only useful as start point.

The exponential problem complexity vs. population size relation has been attributed to the fact that multi-objective optimizers need to have a good sampling of the decision space in order to determine the correct search directions. For example, in Pareto-based optimization algorithms it is necessary that a profuse set of dominated and non-dominated solutions in order to have a usable population ranking.

From that, it can be conjectured that a failure scenario is characterized by the homogeneity in the fitness assignment and its corresponding population ranking. Using such homogeneity indicator it could be detected at early stages of the execution if the evolutionary process would have a negative outcome while it could also detect when the population has converged to a solution.

Some previous works, like (Bui et al., 2009; Martí et al., 2007), have exploited the status of the dominance relations in order to establish progress of the algorithm. These approaches have the drawback that does not take into account the diversity of the population. This may cause them to call for an execution stop when the optimization has not actually reached an stagnation stage but it is improving the spread of the solutions.

That is why we proposed a novel progress indicator, called *fitness homogeneity indicator* (FHI) (Martí et al., 2010e). This indicator improves the other previously discussed indicators as it takes into account all possible processes taking place in the population while not requiring an intensive computation as it relies on the fitness values calculated for the individuals. It is also capable of equally detecting success and failure scenarios, hopefully making an early detection of the second case.

The rest of the referenced work proceeds by discussing the theoretical elements needed in the discussion of this work. Here we also briefly present the current state of the stopping criteria issue. After that, we formally introduce FHI and examine its features as a local progress indicator. In order to establish FHI validity a set of experiments are carried out by using it as progress indicator by two evidence gathering processes, one based on Kalman filters and the other on statistical hypothesis testing. Following the experimentation, some conclusive remarks are put forward.

As mentioned above, we are interested on capturing the degree at which a population receives a well-spread fitness assignment or if all the population receives a similar fitness value, something that can be related to stagnation, due to successful convergence or failure.

There are many possible formulations to gauge this homogeneity. We have selected for our study a simple and computationally inexpensive method: to compute the standard deviation of the fitness values.

A new MGBM stopping criterion can be put forward by combining the recursive estimation prediction/update framework proposed by the MBGM criterion (see Chapters 5 and 6) and FHI.

In order to establish the validity and viability of our proposal we set up an experiment that contrasts FHI with alternative approaches. Among those approaches we can find it predecessor, MBGM, that used the MDR indicator, and other binary quality indicators formulated for MOP solvers. Among those indicators we have chosen two popular ones: hypervolume and the additive epsilon indicator (Knowles et al., 2006a). Similarly we have chosen a statistical hypothesis test to establish FHI validity from a more statistically robust point of view. For that task, a one-sample t-test was used for evidence gathering.

From these experiments we have deduced it is a good starting point for research in this direction. Obviously, more experimentation is required and other types of filters must be tested. Perhaps some work must be done in order to create the necessary assessment tools to be able to gauge the performance of the criteria.

A

Published Results

THIS appendix lists the scientific works that have to do with this thesis organized by class of publication.

Articles

- Martí, L., García, J., Berlanga, A., Coello Coello, C. A., and Molina, J. M. (2011). MB-GNG: Addressing drawbacks in multi-objective optimization estimation of distribution algorithms. *Operations Research Letters*, doi: 10.1016/j.orl.2011.01.002, in press (available online on February 1st, 2011).
- Martí, L., García, J., Berlanga, A., Coello Coello, C. A., and Molina, J. M. (2011). On current model-building methods for multi-objective estimation of distribution algorithms: Shortcommings and directions for improvement. *IEEE Transactions on Evolutionary Computation*, under review.
- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2011). MONEDA: Scalable multi-objective optimization with a neural network–based estimation of distribution algorithm. *Evolutionary Computation*, under review.
- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2011). A stopping criterion for multi-objective optimization evolutionary algorithms. *Evolutionary Computation*, under review.

Technical Reports

- Martí, L., García, J., Berlanga, A., Coello Coello, C. A., and Molina, J. M. (2010). On current model–building methods for multi–objective estimation of distribution algorithms: Shortcommings and directions for improvement. Technical Report GIAA/2010–E001, Grupo de Inteligencia Artificial Aplicada, Universidad Carlos III de Madrid, Colmenarejo, Spain.
- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010). MONEDA: Scalable multi–objective optimization with a neural network–based estimation of distribution

algorithm. Technical Report GIAA/2010–E002, Grupo de Inteligencia Artificial Aplicada, Universidad Carlos III de Madrid, Colmenarejo, Spain.

Book Chapters

- Pérez, M. J., García, J., Martí, L., and Molina, J. M. (2006). Multi–objective optimization evolutionary algorithms in insurance linked derivatives. In Rennard, J.-P., editor, *Handbook of Research on Nature–inspired Computing for Economics and Management*, volume II, pages 885–908. Idea Group, London.

Conferece Proceedings

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010). Moving away from error-based learning in multi-objective estimation of distribution algorithms. In *GECCO '10: Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 545–546, New York, NY, USA. ACM Press.
- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010). Advancing model–building for many–objective optimization estimation of distribution algorithms. In Chio, C. D., Cagnoni, S., Cotta, C., Ebner, M., Ekárt, A., Esparcia-Alcazar, A. I., Goh, C.-K., Merelo, J. J., Neri, F., Preuß, M., Togelius, J., and Yannakakis, G. N., editors, *Applications of Evolutionary Computation*, volume 6024 of *Lecture Notes in Computer Science*, pages 512–521. Springer.
- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010). A progress indicator for detecting success and failure in evolutionary multi–objective optimization. In *2010 IEEE Conference on Evolutionary Computation (CEC), part of 2010 IEEE World Congress on Computational Intelligence (WCCI 2010)*, Piscataway, New Jersey. IEEE Press.
- Guerrero, J. L., Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010). Introducing a robust and efficient stopping criterion for MOEAs. In *2010 IEEE Conference on Evolutionary Computation (CEC), part of 2010 IEEE World Congress on Computational Intelligence (WCCI 2010)*, Piscataway, New Jersey. IEEE Press.
- Fonseca, C., Gandibleux, X., Korhonen, P., Martí, L., Naujoks, B., Thiele, L., Wallenius, J., and Zitzler, E. (2009). 09041 working group on emo for interactive multiobjective optimization (1st round). In Deb, K., Greco, S., Miettinen, K., and Zitzler, E., editors, *Hybrid and Robust Approaches to Multiobjective Optimization*, number 09041 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany. Schloss Dagstuhl — Leibniz–Zentrum fuer Informatik, Germany.
- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009). On the computational properties of the multi–objective neural estimation of distribution algorithms. In Krasnogor, N., Melián-Batista, B., Moreno-Pérez, J. A., Moreno-Vega, J. M., and Pelta, D., editors, *Nature Inspired Cooperative Strategies for Optimization (NICSO 2008)*, Studies in Computational Intelligence, Berlin/Heidelberg. Springer.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009). An approach to stopping criteria for multi–objective optimization evolutionary algorithms: The MGBM criterion. In *2009 IEEE Conference on Evolutionary Computation (CEC 2009)*, pages 1263–1270, Piscataway, New Jersey. IEEE Press.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009). On the model–building issue of multi–objective estimation of distribution algorithms. In *4th International Conference on Hybrid Artificial Intelligence (HAIS'09)*, Lecture Notes in Artificial Intelligence, pages 293–300, Berlin/Heidelberg. Springer.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009). Solving complex high–dimensional problems with the multi–objective neural estimation of distribution algorithm. In Thierens, D., Deb, K., Pelikan, M., Beyer, H.-G., Doerr, B., Poli, R., and Bittari, M., editors, *GECCO 2009: 11th Annual Conference on Genetic and Evolutionary Computation*, pages 619–626, New York, NY, USA. ACM Press.

- Guerrero, J. L., García, J., Martí, L., Molina, J. M., and Berlanga, A. (2009). A stopping criterion based on Kalman estimation techniques with several progress indicators. In Thierens, D., Deb, K., Pelikan, M., Beyer, H.-G., Doerr, B., Poli, R., and Bittari, M., editors, *GECCO 2009: 11th Annual Conference on Genetic and Evolutionary Computation*, pages 587–594, New York, NY, USA. ACM Press.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2008). Introducing MONEDA: Scalable multiobjective optimization with a neural estimation of distribution algorithm. In Thierens, D., Deb, K., Pelikan, M., Beyer, H.-G., Doerr, B., Poli, R., and Bittari, M., editors, *GECCO '08: 10th Annual Conference on Genetic and Evolutionary Computation*, pages 689–696, New York, NY, USA. ACM Press. *EMO Track "Best Paper" Nominee*.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2008). Model–building algorithms for multiobjective EDAs: Directions for improvement. In Michalewicz, Z., editor, *2008 IEEE Conference on Evolutionary Computation (CEC), part of 2008 IEEE World Congress on Computational Intelligence (WCCI 2008)*, pages 2848–2855, Piscataway, New Jersey. IEEE Press.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2008). Scalable continuous multiobjective optimization with a neural network–based estimation of distribution algorithm. In Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G. A., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A. I., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A. c., and Yang, S., editors, *Applications of Evolutionary Computing*, volume 4974 of *Lecture Notes in Computer Science*, pages 535–544. Springer, Heidelberg.

- Martí, L. (2007). A hybrid neural system based on adaptive resonance theory and representational redescription capable of variable binding. In Si, J. and Sun, R., editors, *2007 International Joint Conference on Neural Networks (IJCNN)*, pages 2448–2453. IEEE Press.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2007). A cumulative evidential stopping criterion for multiobjective optimization evolutionary algorithms. In Thierens,

D., Deb, K., Pelikan, M., Beyer, H.-G., Doerr, B., Poli, R., and Bittari, M., editors, *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO '07)*, page 911, New York. ACM Press.

- Martí, L., García, J., Berlanga, A., and Molina, J. M. (2007). A cumulative evidential stopping criterion for multiobjective optimization evolutionary algorithms. In *GECCO'07: Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation*, pages 2835–2842, New York, NY, USA. ACM.

B

Statistical Interpretation of Experiments

THE stochastic nature of evolutionary algorithms prompts the use of statistical tools in order to reach a valid judgement of the quality of the solutions and how different algorithms compare with each other.

The straightforward approach to experiment design is to run the algorithm for a given number of independent executions and then extract some descriptive statistics of the results, for example, the means and standard deviations. As the analysis and contrasting of the plain numbers can be a rather burdensome and exhaustive task some graphical representations should be used.

Box plots (Benjamini, 1988; Chambers et al., 1983) are one of such representations and have been repeatedly applied in our context. A box plot produces a box and whisker plot for each set of observations (in our case indicator values). The box represents the lower and upper quartile values, generally the 25^{th} and 75^{th}, respectively. A line inside the box marks the median and the notches serve to roughly gauge confidence level of this median. The whiskers are lines extending from each end of the box to show the extent of the rest of the data. Outliers are data is with values beyond the ends of the whiskers. Figure B.1 serves as an explanatory example of the outcome of a comparative experiment being represented as a box plot.

Box plots allows a visual comparison of the results and, in principle, some conclusions could be deduced out of them. Nevertheless, in order to reach a substantiated judgement it is necessary go beyond reporting the descriptive statistics of the performance indicators. For this task is required to carry out a set of statistical inferences that would support any judgements made from the data.

A common way of carrying out these inferences is through a set of techniques grouped under the term statistical hypothesis test (Conover, 1999). A hypothesis test is the process of inferring if a certain statement about a phenomenon appear to be true or false by using sample data.

In these tests, the statement that is to be verified is posed as the *alternate hypothesis*. The complementary hypothesis that would imply a rejection of the previous one is know as the *null hypothesis*. This last hypothesis is the one that is contrasted by the hypothesis test. In

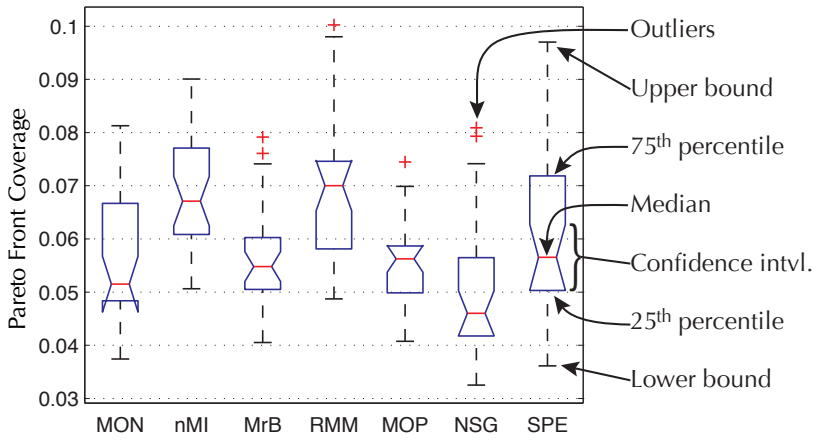


Figure B.1: An example of a box plot that reports the results of an EMO experiment comparing different results. The different parts of the box plots are pointed out.

order to do this a *test statistic* must be selected. This test statistic is supposed to expose the properties of the null hypothesis by taking a certain value when the null hypothesis is true.

The test statistic is used to compute the significance level, α, that is, the probability of rejecting the null hypothesis while it is true. Traditionally, the null hypothesis is rejected if the significance level is smaller than or equal to a given threshold known as p-value. For example, if the level is set to 0.05, then results that are only 5% likely or less, given that the null hypothesis is true, are deemed extraordinary.

There is a rather broad set of test statistics. They can be grouped in parametric and non-parametric tests. Parametric approaches make more assumptions than non-parametric ones. In particular, they frequently assume that data follows a given probability distribution. If those extra assumptions are correct, parametric methods can produce more accurate and precise estimates. However, if those assumptions are incorrect, parametric methods can be misleading.

In the case of assessing MOP optimizers performance assessment, the use of parametric methods is not advised since the distribution of the data is unknown. Furthermore, as sample sizes are reduced due to of the high computational cost of obtaining them, the large numbers law (Bernoulli, 1713) can not be applied. In contrast, non-parametric methods do not presuppose anything about the data being analyzed. Consequently, these are the approaches most commonly used in context of interest of this thesis.

The core and remaining issue is to correctly formulate the test hypothesis, and, selecting the correct test statistic. Different frameworks for carrying out this task have been extensively discussed by other authors (see, for example, Coello Coello et al. (2007); Knowles et al. (2006a); Zitzler et al. (2008)). At first glance, when comparing the performance of two or more algorithms it would be interesting to assert which algorithm has a better mean of its indicator values. However this is not enough to establish the outperformance of one algorithm with regard to the others. What should be determine is if two algorithms have yield statistically significant different results or not, or, in other words, if they were produced by the same probability distribution or not.

A suitable way of determining this is by applying the Kruskal-Wallis test (Kruskal and Wallis, 1952) with the indicator values yielded by each algorithm's run. In this experimentation context, the test had as its null hypothesis that all algorithms were equally capable of solving the problem.

Expressing it more formally, for a set of algorithms $A_1,\ldots, A_K$, each one is run r times on the same problem. Let $I_{k,j}$ be the value of the indicator yielded by algorithm k in run j and $N = rK$.

In our particular problem, the Kruskal–Wallis test goes on by sorting $I_{k,j}$, by relying on $R(I_{k,j})$, the ranking function that returns the position of measurement $I_{k,j}$ in the list. Following that, the rank sum is calculated for each algorithm,

$$R_k = \sum_{j=1}^{r} R\left(I_{k,j}\right);\ k = 1,\ldots,K\,. \tag{B.1}$$

The test statistic T is then computed as

$$T = \frac{1}{S^2}\left[\left(\frac{1}{r}\sum_{k=1}^{K} R_i^2\right) - \frac{N\left(N+1\right)^2}{4}\right], \tag{B.2}$$

with

$$S^2 = \frac{1}{N-1}\left(\left(\sum_{k=1}^{K}\sum_{j=1}^{r} R\left(I_{k,j}\right)\right) - \frac{N\left(N+1\right)^2}{4}\right). \tag{B.3}$$

If T is greater than the $1-\alpha$ quantile of the χ^2 distribution with $K-1$ degrees of freedom (Wilson and Hilferty, 1931) the null hypothesis is accepted at significance level α. In this case it means that at least one of the algorithms generated significantly different than at least one of other algorithms.

In case that the null hypothesis is rejected (a situation that, for example, happened in all our experiments) the Conover–Inman procedure (Conover, 1999, pp.288-290) can be applied in a pairwise manner in order to determine if the results of one algorithm were significantly better than those of the other. In particular, the difference of indicator values yielded by algorithms A_k and A_h is statistically significant if

$$\left|\frac{R_k - R_h}{r}\right| > t_{1-\alpha/2} S \sqrt{\frac{2\left(N-1-T\right)}{r\left(N-K\right)}}, \tag{B.4}$$

where $t_{1-\alpha/2}$ is the $1-\alpha/2$ quantile of the Student's t distribution (Gosset a.k.a. Student, 1908). A similar test framework has been previously applied for assessing similar experiments (Bader, 2010; Bader and Zitzler, 2011).

There are some possible alternatives, like the use of the Mann–Whitney–Wilcoxon U test (Mann and Whitney, 1947; Wilcoxon, 1945) or the Kolmogorov–Sminoff test (Massey, 1951). We decided apply the previously described test methodology as it has been successfully used before in the multi-objective experimental contexts.

B.1 Measuring the Algorithms Computational Costs

Besides measuring how good the solutions obtained from the algorithms are, it is also very important to understand how big the computational effort required to reach those solutions is. This effort is expressed in two ways: spatial and temporal. The first refers to the amount of storage space (memory, disk, etc.) used by an algorithm during the optimization process. The second deals with the time consumed by the algorithm in order to reach the solution. This last quantity is the one of interest in this work as it is the most critical given the current state of computing technology.

Because of the stochastic nature of evolutionary approaches, the traditional methods for estimating the computational complexity of algorithms are no longer suitable. That is why different alternative strategies for the assessing the temporal complexity of multi-objective optimizers have been put forward.

The simplest one is to measure the time employed in each independent run and then obtaining a mean execution time. This procedure is sensitive to the uncontrollable influence of concurrent hardware and software processes like memory swapping, garbage collection, etc., that might interfere with an accurate measurement.

A more common approach is to compute the number of algorithm iterations (the number of generations in the evolutionary case) needed for reaching the results. This method has the advantage of providing a measurement that is repeatable using different combinations of hardware and software. On the downside, it does not account for the time consumed in carrying out each iteration. This intra-iteration time is often considerable; therefore disregarding it may lead to improper conclusions.

The third strategy counts the number of evaluations of the objective functions. This method is rooted in real-life engineering problems where evaluations are usually costly and should be minimized. This approach provides more complete information than the previous ones. In spite of that, it does not take the amount of computation dedicated to the optimization process itself into account, which can be the most time demanding parts.

A form of having a better understanding on the time complexity is to measure the number of floating-point operations carried out by each algorithm. This approach assumes that all floating-point operations have to do with the optimization process itself. This requirement can be easily met under experimental conditions.

There are a number of profiling tools that are capable of tracking the number of floating-point operations that have taken place as part of a process. For this work we have chosen the OProfile program profiling toolkit (Levon, 2004).

C

Multi–Objective Test Problems

THIS appendix describes the test problems used in the experiments of the thesis. There are now a rather large number of test problems for multi-objective optimization (see (Huband et al., 2006b) for a comprehensive review). For this thesis we have selected problems taken from the DTLZ and WFG problems sets. They are described bellow.

C.1 The DTLZ Problem Set

The DTLZ3, DTLZ6 and DTLZ7 problems are part of the Deb–Thiele–Laumann–Zitzler (DTLZ) family of scalable multi-objective test problems (Deb et al., 2004). They were introduced to study and compare the performance of different MOEAs in high-dimensional (many-objective) situations.

C.1.1 DTLZ3

The DTLZ3 problem is a M-objective problem with a n-dimensional decision vector based on the DTLZ2. The Pareto-optimal front lies on the first orthant of a unit hypersphere (see figure C.1a for a three-dimensional representation). This problem was introduced to test the ability of a MOEA to converge to the global Pareto–optimal front, since there are $3^{n-M+1}-1$ suboptimal fronts parallel to the optimal one. It follows the specification introduced in (2.1) with the objective functions formulated as

$$\begin{aligned} f_1(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\prod_{i=1}^{M-1}\cos(x_i\tfrac{\pi}{2}), \\ &\vdots \\ f_m(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\prod_{i=1}^{M-m}\cos(x_i\tfrac{\pi}{2})\sin(x_{M-m+1}\tfrac{\pi}{2}), \\ &\vdots \\ f_M(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\sin(x_1\tfrac{\pi}{2}) \end{aligned}$$

having $g(\boldsymbol{x}_M)$ defined as

$$g(\boldsymbol{x}_M) = 100\left[|\boldsymbol{x}_M| + \sum_{x_i\in\boldsymbol{x}_M}(x_i-0.5)^2 - \cos 20\pi(x_i-0.5)\right],$$

where $\boldsymbol{x}_M$ represents the last $n-M+1$ features of $\boldsymbol{x}\in[0,1]^n$.

C.1.2 DTLZ6

The DTLZ6 problem is also based on a simpler problem, in this case the DTLZ5 problem. As in the previous case, suboptimal fronts are also present with the intention of deceiving the optimizer.

The objective functions are expressed as

$$\begin{aligned} f_1(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\prod_{i=1}^{M-1}\cos(\theta_i\tfrac{\pi}{2}),\\ &\vdots\\ f_m(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\prod_{i=1}^{M-m}\cos(\theta_i\tfrac{\pi}{2})\sin(\theta_{M-m+1}\tfrac{\pi}{2}),\\ &\vdots\\ f_M(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\sin(\theta_1\tfrac{\pi}{2}) \end{aligned}$$

with $g(\boldsymbol{x}_M)$ defined as

$$g(\boldsymbol{x}_M) = \sum_{x_i\in\boldsymbol{x}_M} x_i^{0.1};$$

and $\theta_1,\ldots,\theta_{M-1}$ as

$$\begin{aligned} \theta_1 &= x_1\tfrac{\pi}{2}\\ \theta_i &= \tfrac{\pi}{4(1+g(\boldsymbol{x}_M))}(1+2g(\boldsymbol{x}_M)x_i) \end{aligned}$$

The Pareto-optimal front corresponds to $x_i = 0$ for $x_i \in \boldsymbol{x}_M$. A graphical representation can be examined on figure C.1b.

C.1.3 DTLZ7

The DTLZ7 problem has a Pareto-optimal front that consists of a heavily disconnected set of 2^{M-1} Pareto-optimal regions. This problem is intended to test an algorithm's ability to maintain a robust coverage of all optimal regions. It is formulated as

$$\begin{aligned} f_m(\boldsymbol{x}) &= x_m, \text{ for } m = 1,\ldots,M-1;\\ f_M(\boldsymbol{x}) &= (1+g(\boldsymbol{x}_M))\left[M-\sum_{i=1}^{M-1}\tfrac{f_i}{1+g(\boldsymbol{x}_M)}(1+\sin 3\pi f_i)\right] \end{aligned}$$

with g defined as

$$g = 1+\frac{9}{|\boldsymbol{x}_M|}\sum_{x_i\in\boldsymbol{x}_M} x_i.$$

The Pareto-optimal front corresponds to $x_i = 0$ for $x_i \in \boldsymbol{x}_M$. A 3D graphical representation of it is presented in figure C.1c.

C.2 The Walking Fish Group Problem Set

The problems to be addressed are part of the Walking Fish Group problem toolkit (WFG) (Huband et al., 2006b). This is a toolkit for creating complex synthetic multi-objective test problems that can be devised to exhibit a given set of target features.

Unlike previous test suites where complexity is embedded in the problem, a test problem designer using the WFG toolkit has access to a series of components to control specific test problem features (e.g., separability, modality, etc.). The WFG toolkit was used to construct a suite of test problems that provides a thorough test for optimizers. This set of nine problems, WFG1–WFG9, are formulated in such manner that each poses a different type of challenge to multi-objective optimizers.

The WFG test suite exceeds the functionality of previous existing test suites. In particular, it includes a number of problems that exhibit properties not evident in other commonly used test suites such as the DTLZ and the Zitzler–Deb–Thiele (ZDT) (Zitzler et al., 2000) test suites. These differences include: non-separable problems, deceptive problems, a truly degenerate problem, a mixed shape Pareto front problem, problems scalable by the number of position-related parameters, and problems with dependencies between position- and distance-related parameters. The WFG test suite provides a better form of assessing the performance of optimization algorithms on a wide range of different problems.

From the set of nine problems, the test functions WFG4 to WFG9 were selected in Chapter 8 because of the simple form of their Pareto-optimal fronts, which lie on the first orthant of a unit hypersphere. For this reason, the progress of the optimization process can be determined without having a sampled version of the Pareto-optimal front.

WFG problems are constructed by combining functions that define the shape of the Pareto-optimal front and a set of transformation functions. The shape functions are:

$$
\begin{aligned}
\text{linear}_1(x_1,\ldots,x_{M-1}) &= \prod_{i=1}^{M-1} x_i\,; \\
\text{linear}_{m=2,\ldots,M-1}(x_1,\ldots,x_{M-1}) &= \left(\prod_{i=1}^{M-m} x_i\right)(1-x_{M-m+1})\,; \\
\text{linear}_M(x_1,\ldots,x_{M-1}) &= 1-x_1\,; \\
\text{convex}_1(x_1,\ldots,x_{M-1}) &= \prod_{i=1}^{M-1}\left(1-\cos\left(x_i\tfrac{\pi}{2}\right)\right)\,; \\
\text{convex}_{m=2,\ldots,M-1}(x_1,\ldots,x_{M-1}) &= \left[\prod_{i=1}^{M-1}\left(1-\cos\left(x_i\tfrac{\pi}{2}\right)\right)\right]\left(1-\sin\left(x_{M-m+1}\tfrac{\pi}{2}\right)\right)\,; \\
\text{convex}_M(x_1,\ldots,x_{M-1}) &= \left(1-\sin\left(x_{M-m+1}\tfrac{\pi}{2}\right)\right)\,; \\
\text{concave}_1(x_1,\ldots,x_{M-1}) &= \prod_{i=1}^{M-1}\left(1-\sin\left(x_i\tfrac{\pi}{2}\right)\right)\,; \\
\text{concave}_{m=2,\ldots,M-1}(x_1,\ldots,x_{M-1}) &= \left[\prod_{i=1}^{M-1}\left(1-\sin\left(x_i\tfrac{\pi}{2}\right)\right)\right]\left(1-\cos\left(x_{M-m+1}\tfrac{\pi}{2}\right)\right)\,; \\
\text{concave}_M(x_1,\ldots,x_{M-1}) &= \cos\left(x_{M-m+1}\tfrac{\pi}{2}\right)\,; \\
\text{mixed}_M(x_1,\ldots,x_{M-1}) &= \left(1-x_1-\frac{\cos 2A\pi x_1+\pi/2}{2A\pi}\right)^{\alpha}\,; \\
\text{disc}_M(x_1,\ldots,x_{M-1}) &= 1-x_1^{\alpha}\cos^2\left(Ax_1^{\beta}\pi\right)\,.
\end{aligned}
$$

Similarly, the transformation functions are formulated as:

$$
\begin{aligned}
\mathrm{bPoly}(y,\alpha) =& y^{\alpha}\,;\\
\mathrm{bFlat}(y,A,B,C) =& A+\min\left(0,|y-B|\right)\frac{A(B-y)}{B}-\min\left(0,|y-C|\right)\frac{(1-A)(y-C)}{1-C}\,;\\
\mathrm{bParam}(y,u(\boldsymbol{y}'),A,B,C) =& y^{B+(C-B)(A-(1-2u(\boldsymbol{y}'))|\lfloor 0.5-u(\boldsymbol{y}')\rfloor+A|}\,;\\
\mathrm{sLinear}(y,A) =& \frac{|y-A|}{|\lfloor A-y\rfloor+A|}\,;\\
\mathrm{sDecept}(y,A,B,C) =& 1+(|y-A|-B)\\
& \left(\frac{\lfloor y-A+B\rfloor\left(1-C^{\frac{A-B}{B}}\right)}{A-B}+\frac{\lfloor A+B-y\rfloor\left(1-C^{\frac{1-A-B}{B}}\right)}{1-A-B}+\frac{1}{B}\right)\,;\\
\mathrm{sMulti}(y,A,B,C) =& \frac{1+\cos\left[(4A+2)\,\pi\left(0.5-\frac{|y-C|}{2(\lfloor C-y\rfloor+C)}\right)\right]+4B\left(\frac{|y-C|}{2(\lfloor C-y\rfloor+C)}\right)}{B+2}\,;\\
\mathrm{rSum}(\boldsymbol{y},\boldsymbol{w}) =& \frac{\sum_{i=1}^{|\boldsymbol{y}|}w_1y_i}{\sum_{i=1}^{|\boldsymbol{y}|}w_i}\,;\\
\mathrm{rNonSep}(\boldsymbol{y},A) =& \frac{\sum_{j=1}^{|\boldsymbol{y}|}\left(y_j+\sum_{k=0}^{A-2}\left|y_j-y_{1+j+k\mod|\boldsymbol{y}|}\right|\right)}{\frac{|\boldsymbol{y}|}{A}\lceil\frac{A}{2}\rceil\left(1+2A-2\lceil\frac{A}{2}\rceil\right)}\,.
\end{aligned}
$$

There are some other common features for all problems. For example, their decision vector is

$$\boldsymbol{z}=[z_1,\ldots,z_k,z_{k+1},\ldots,z_n]\,,0\le z_i\le z_{i,\max}\,.$$

and

$$
\begin{aligned}
z_{i=1:n,\max} &= 2i\,;\\
z_{i=1:n,[0,1]} &= \frac{z_i}{z_{i=1:n,\max}}\,;\\
x_{i=1:M-1} &= \max(y_M,A_i)(y_i-0.5)+0.5\,;\\
x_M &= y_M\,;\\
S_{m=1:M} &= 2m\,;\\
A_{i=1:M} &= 1\,.
\end{aligned}
$$

C.2.1 WFG1

WFG1 skews the relative significance of different parameters by employing dissimilar weights in its weighted sum reduction. It is separable and unimodal.

$$
\begin{aligned}
\text{minimize } f_m(\boldsymbol{x}) &= x_M+S_m\mathrm{convex}_m(x_1,\ldots,x_{M-1});\ m=1,\ldots,M-1;\\
f_M(\boldsymbol{x}) &= x_M+S_M\mathrm{mixed}_M(x_1,\ldots,x_{M-1}),\ \alpha=1,\ A=5\,.
\end{aligned}
$$

where

$$\begin{aligned}
y_{i=1:M-1} &= \text{rSum}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, \quad y'_{ik/(M-1)}\right], \right. \\
&\qquad \left.[2\left((i-1)k(M-1)+1\right), \ldots, 2ik/(M-1)]\right); \\
y_M &= \text{rSum}\left(\left[y'_{k+1}, \ldots, y'_n\right], [2(k+1), \ldots, 2n]\right); \\
y'_{i=1:n} &= \text{bPoly}(y''_i, 0.02); \\
y''_{i=1:k} &= y'''_i; \\
y''_{i=k+1:n} &= \text{bFlat}(y'''_i, 0.8, 0.75, 0.85); \\
y'''_{i=1:k} &= z_{i,[0,1]}; \\
y'''_{i=k+1:n} &= \text{sLinear}(z_{i,[0,1]}, 0.35).
\end{aligned}$$

C.2.2 WFG2

This is a non-separable problem with a disconnected Pareto-optimal front.

$$\begin{aligned}
\text{minimize } f_m(\boldsymbol{x}) &= x_M + S_m \text{convex}_m(x_1, \ldots, x_{M-1}),\ m = 1, \ldots, M-1; \\
f_M(\boldsymbol{x}) &= x_M + S_M \text{disc}_M(x_1, \ldots, x_{M-1}),\ \alpha = \beta = 1,\ A = 5.
\end{aligned}$$

where

$$\begin{aligned}
y_{i=1:M-1} &= \text{rSum}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], \mathbf{1}\right); \\
y_M &= \text{rSum}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], \mathbf{1}\right); \\
y'_{i=1:k} &= y''_i; \\
y'_{i=k+1:k+l/2} &= \text{rNonSep}\left(\left[y''_{k+2(i-k)-1}, y''_{k+2(i-k)}\right], 2\right); \\
y''_{i=1:k} &= z_{i,[0,1]}; \\
y''_{i=k+1:n} &= \text{sLinear}(z_{i,[0,1]}, 0.35).
\end{aligned}$$

C.2.3 WFG4

WFG4 is a separable and strongly multi-modal problem that, like the remaining problems, has a concave Pareto-optimal front. This front lies on the first orthant of a hypersphere of radius one located at the origin.

$$\text{minimize } f_m = x_M + S_m \text{concave}_m(x_1, \ldots, x_{M-1}),\ m = 1, \ldots, M-1;$$

where

$$\begin{aligned} y_{1:M-1} &= \text{rSum}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], \mathbf{1}\right); \\ y_M &= \text{rSum}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], \mathbf{1}\right); \\ y'_{i=1:n} &= \text{sMulti}(z_{i,[0,1]}, 30, 10, 0.35). \end{aligned}$$

C.2.4 WFG5

WFG5 is also a separable problem but it has a set of deceptive locally optimal fronts. This feature is meant to evaluate the capacity of the optimizers to avoid getting trapped in local optima.

$$\text{minimize } f_m = x_M + S_m \text{concave}_m(x_1, \ldots, x_{M-1});\ m = 1, \ldots, M-1;$$

where

$$\begin{aligned} y_{1:M-1} &= \text{rSum}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], \mathbf{1}\right); \\ y_M &= \text{rSum}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], \mathbf{1}\right); \\ y'_{i=1:n} &= \text{sDecept}(z_{i,[0,1]}, 0.35, 0.001, 0.05). \end{aligned}$$

C.2.5 WFG6

WFG6 is a non-separable problem without the strong multi-modality of WFG4 but with a simpler non-separable reduction when compared to WFG2.

$$\text{minimize } f_m = x_M + S_m \text{concave}_m(x_1, \ldots, x_{M-1});\ m = 1, \ldots, M-1;$$

where

$$\begin{aligned} y_{1:M-1} &= \text{rNonSep}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], k/(M-1)\right); \\ y_M &= \text{rNonSep}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], l\right); \\ y'_{i=1:k} &= z_{i,[0,1]}; \\ y'_{i=k+1:n} &= \text{sLinear}(z_{i,[0,1]}, 0.35). \end{aligned}$$

C.2.6 WFG7

The WFG7 problem is uni-modal and separable.

$$\text{minimize } f_m = x_M + S_m \text{concave}_m(x_1, \ldots, x_{M-1});\ m = 1, \ldots, M-1;$$

where

$$
\begin{aligned}
y_{1:M-1} &= \text{rSum}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], \mathbf{1}\right) ; \\
y_M &= \text{rSum}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], \mathbf{1}\right) ; \\
y'_{i=1:k} &= y''_i ; \\
y'_{i=k+1:n} &= \text{sLinear}\left(y''_i, 0.35\right) ; \\
y''_{i=1:k} &= \text{bParam}\left(z_{i,[0,1]}, \text{rSum}\left(\left[z_{i+1,[0,1]}, \ldots, z_{n,[0,1]}\right], \mathbf{1}\right), 0.98/49.98, 0.02, 50\right) ; \\
y''_{i=k+1:n} &= z_{i,[0,1]} .
\end{aligned}
$$

C.2.7 WFG8

WFG8 is a non-separable problem.

$$\text{minimize } f_m = x_M + S_m \text{concave}_m(x_1, \ldots, x_{M-1});\ m = 1, \ldots, M-1;$$

where

$$
\begin{aligned}
y_{1:M-1} &= \text{rSum}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], \mathbf{1}\right) ; \\
y_M &= \text{rSum}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], \mathbf{1}\right) ; \\
y'_{i=1:k} &= y''_i ; \\
y'_{i=k+1:n} &= \text{sLinear}\left(y''_i, 0.35\right) ; \\
y''_{i=1:k} &= z_{i,[0,1]} ; \\
y''_{i=k+1:n} &= \text{bParam}\left(z_{i,[0,1]}, \text{rSum}\left(\left[z_{1,[0,1]}, \ldots, z_{i-1,[0,1]}\right], \mathbf{1}\right), 0.98/49.98, 0.02, 50\right) .
\end{aligned}
$$

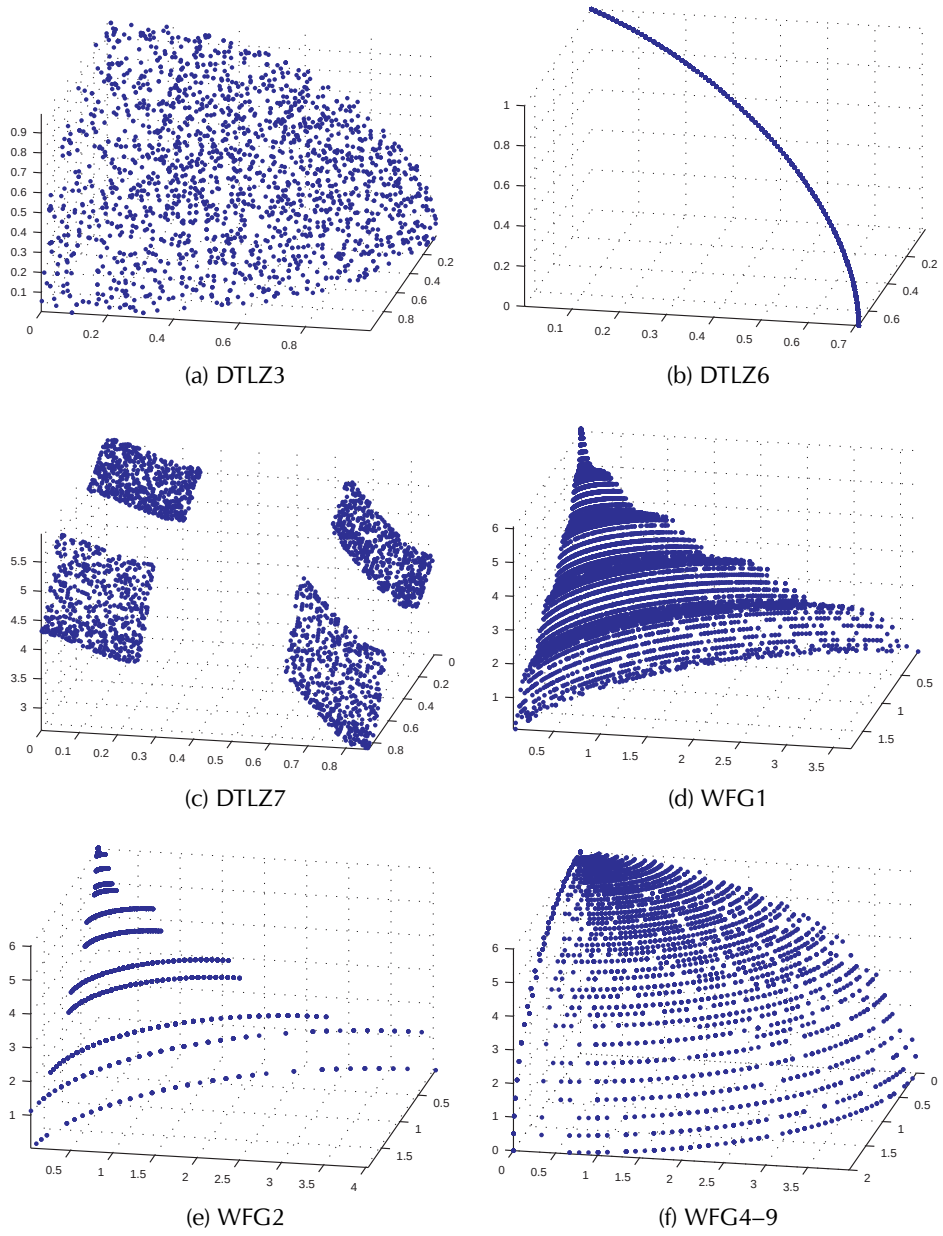
C.2.8 WFG9

WFG9 is non-separable, multi-modal and has deceptive local optima. These properties probably make WFG9 the hardest problem of all the problems of the WFG set.

$$\text{minimize } f_m = x_M + S_m \text{concave}_m(x_1, \ldots, x_{M-1});\ m = 1, \ldots, M-1;$$

where

$$
\begin{aligned}
y_{1:M-1} &= \text{rNonSep}\left(\left[y'_{(i-1)k/(M-1)+1}, \ldots, y'_{ik/(M-1)}\right], k/(M-1)\right) ; \\
y_M &= \text{rNonSep}\left(\left[y'_{k+1}, \ldots, y'k + l/2\right], l\right) ; \\
y'_{i=1:k} &= \text{sDecept}(y''_i, 0.35, 0.001, 0.05) ; \\
y'_{i=k+1:n} &= \text{sMulti}\left(y''_i, 30, 95, 0.35\right) ; \\
y''_{i=1:n-1} &= \text{bParam}\left(z_{i,[0,1]}, \text{rSum}\left(\left[z_{i+1,[0,1]}, \ldots, z_{n,[0,1]}\right], \mathbf{1}\right), 0.98/49.98, 0.02, 50\right) ; \\
y''_{i=n} &= z_{n,[0,1]} .
\end{aligned}
$$



(a) DTLZ3 (b) DTLZ6

(c) DTLZ7 (d) WFG1

(e) WFG2 (f) WFG4–9

Figure C.1: Representation of the Pareto-optimal fronts of DTLZ3, DTLZ6, DTLZ7, WFG1, WFG2 and WFG6 problems configured with three objectives ($M = 3$).

D

MOEDAs Progress In Time

DETAILED results of naïve MIDEA, MrBOA, RM–MEDA, MOPED, NSGA–II, SPEA2 and MONEDA when solving the DTLZ3, DTLZ6, DTLZ7, WFG1, WFG2, and WFG6 problems. For each iteration the mean of the indicators across the 30 runs are plotted. See Chapter 8 for details on the nature of these results.

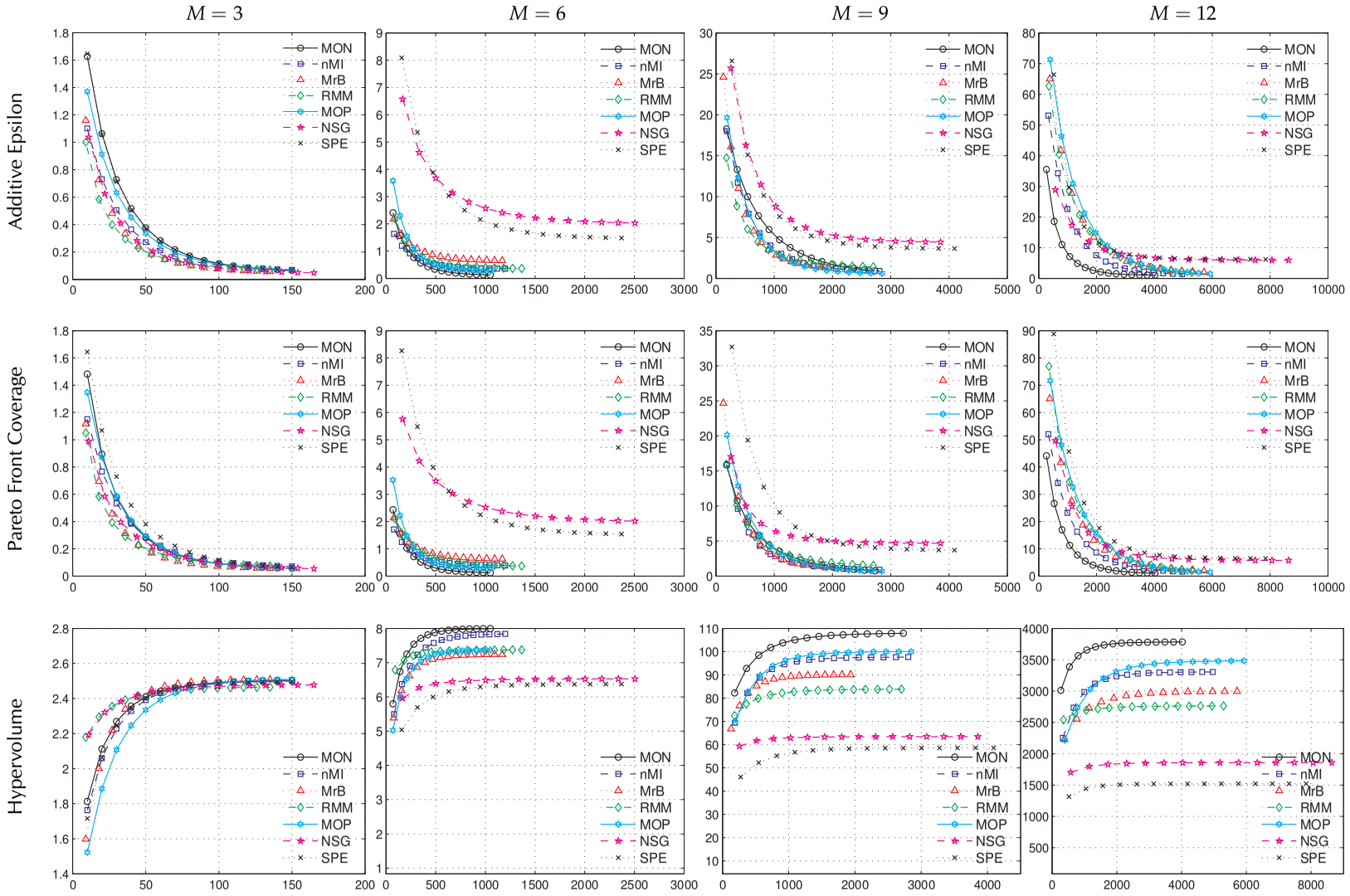


Figure D.1: Algorithms progress according to the different performance indicators while solving the DTLZ3 problem.

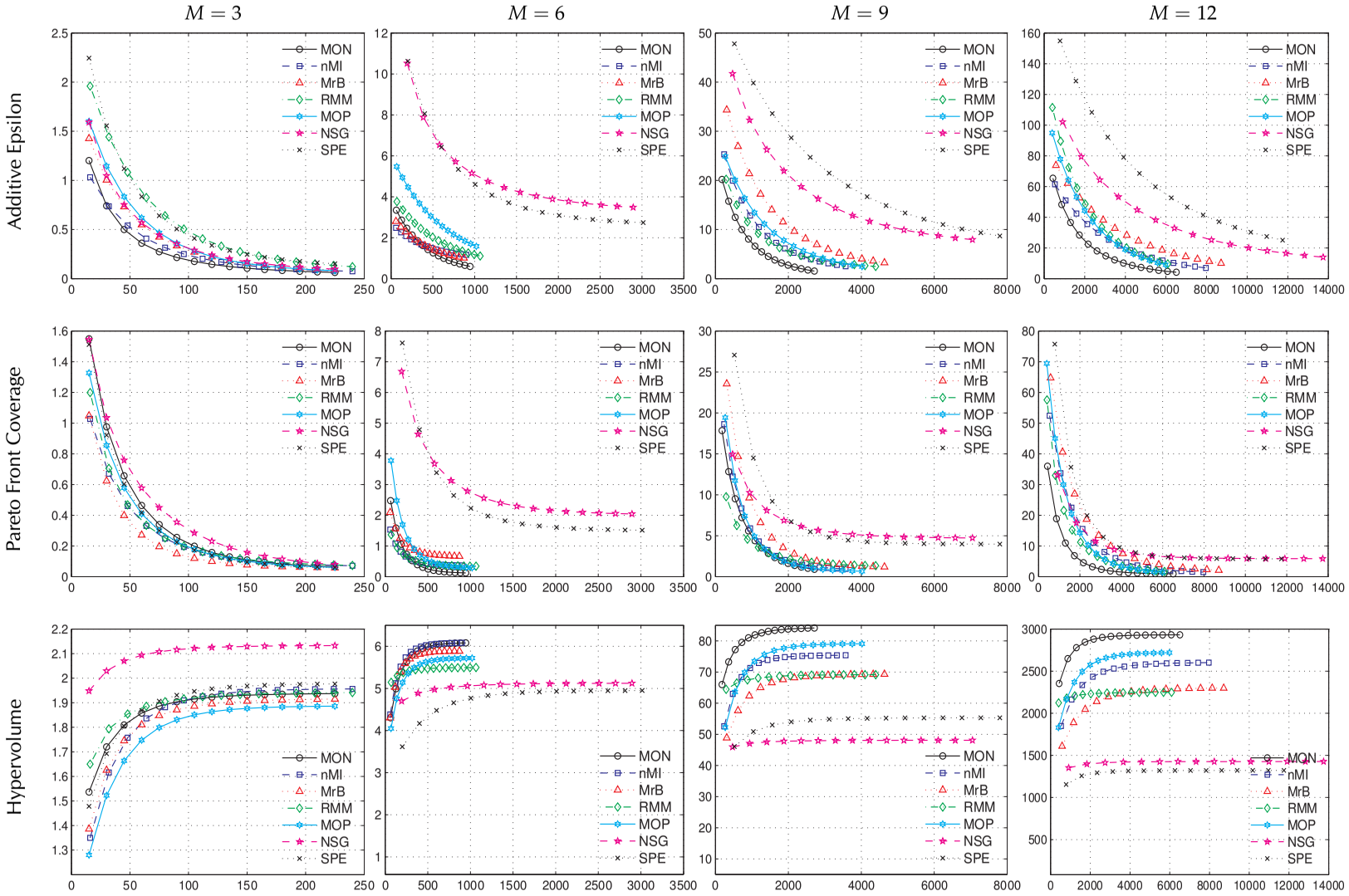


Figure D.2: Algorithms progress according to the different performance indicators while solving the DTLZ6 problem.

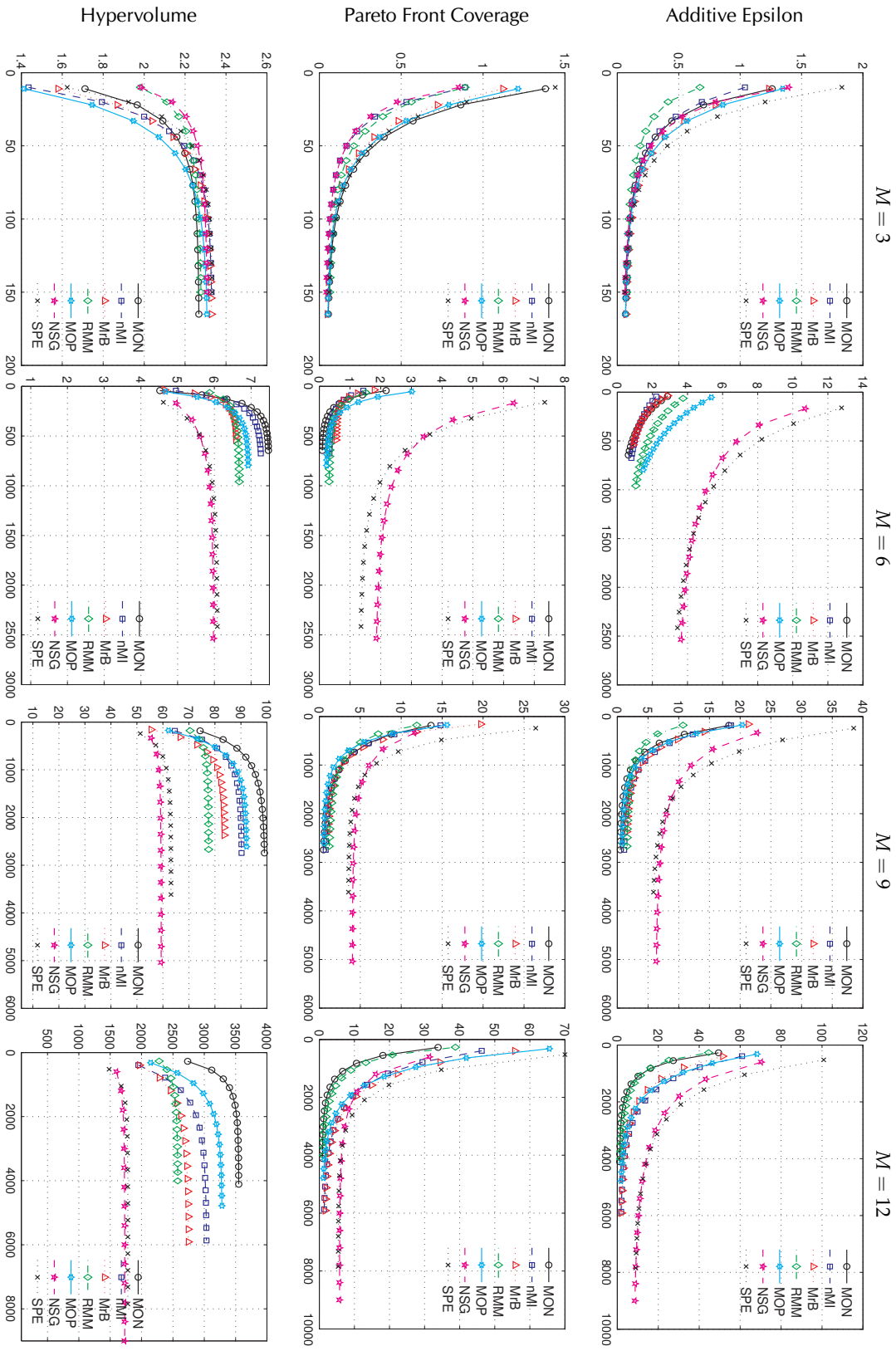


Figure D.3: Algorithms progress according to the different performance indicators while solving the DTLZ7 problem.

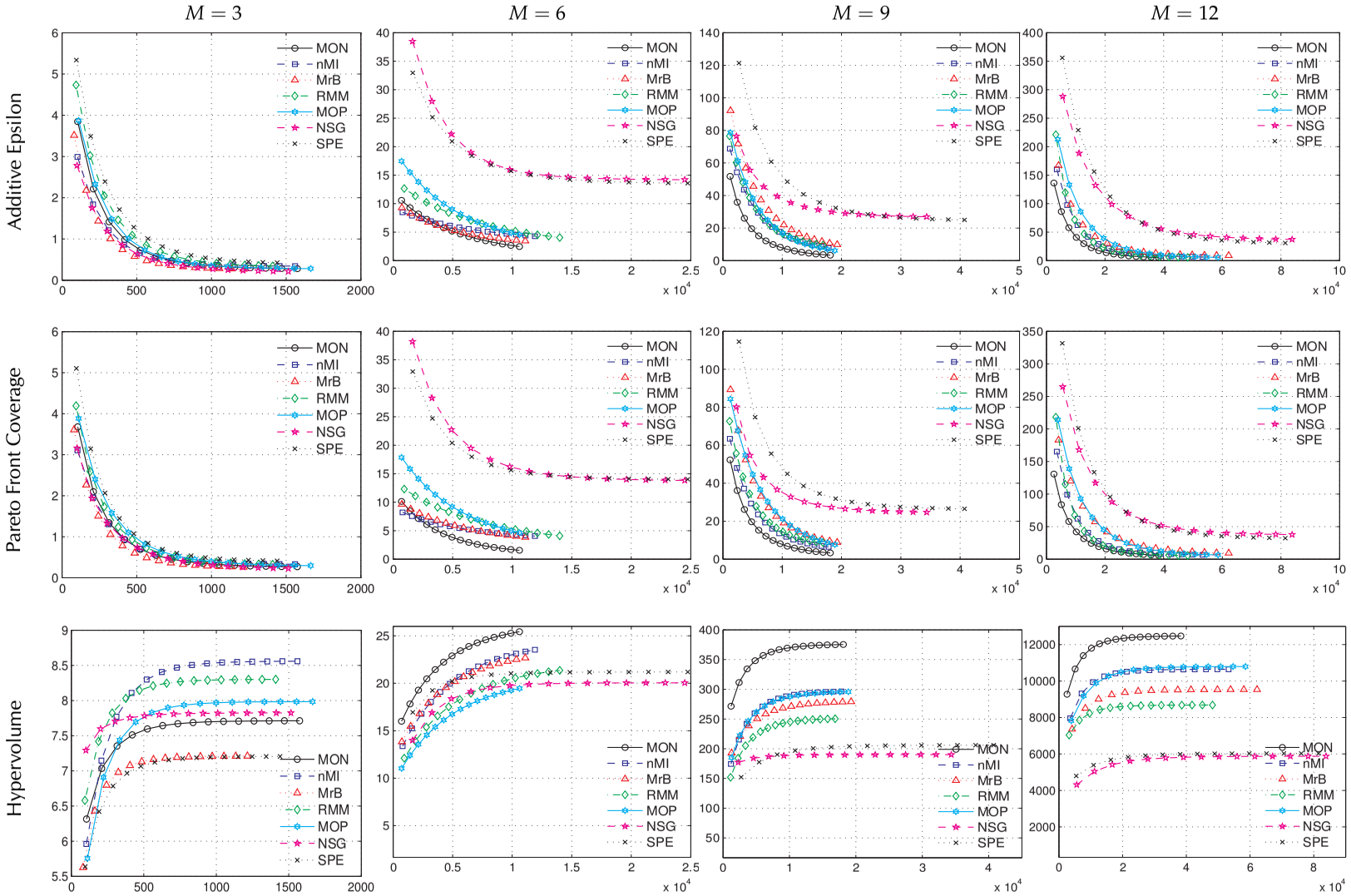


Figure D.4: Algorithms progress according to the different performance indicators while solving the WFG1 problem.

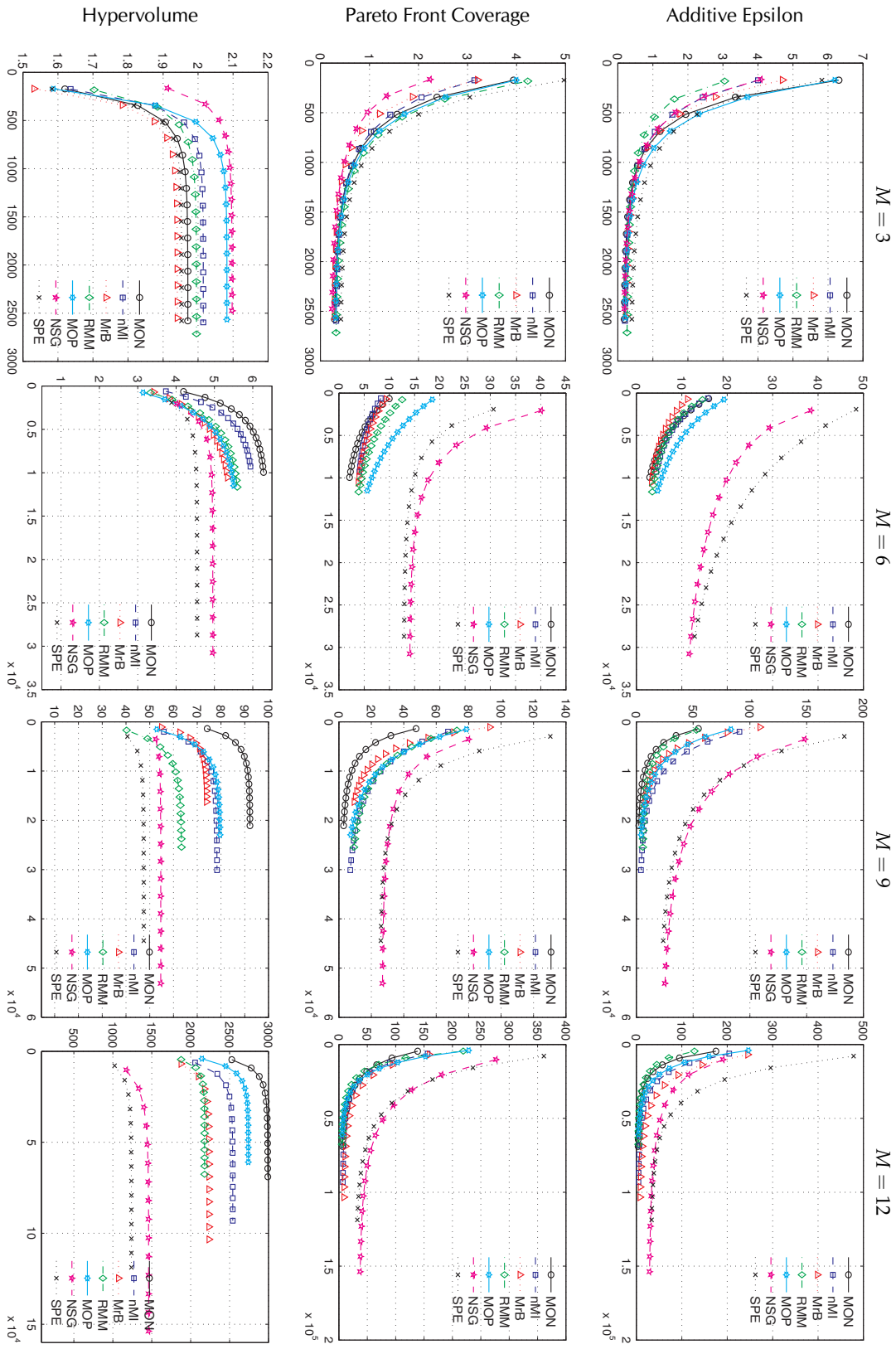


Figure D.5: Algorithms progress according to the different performance indicators while solving the WFG2 problem.

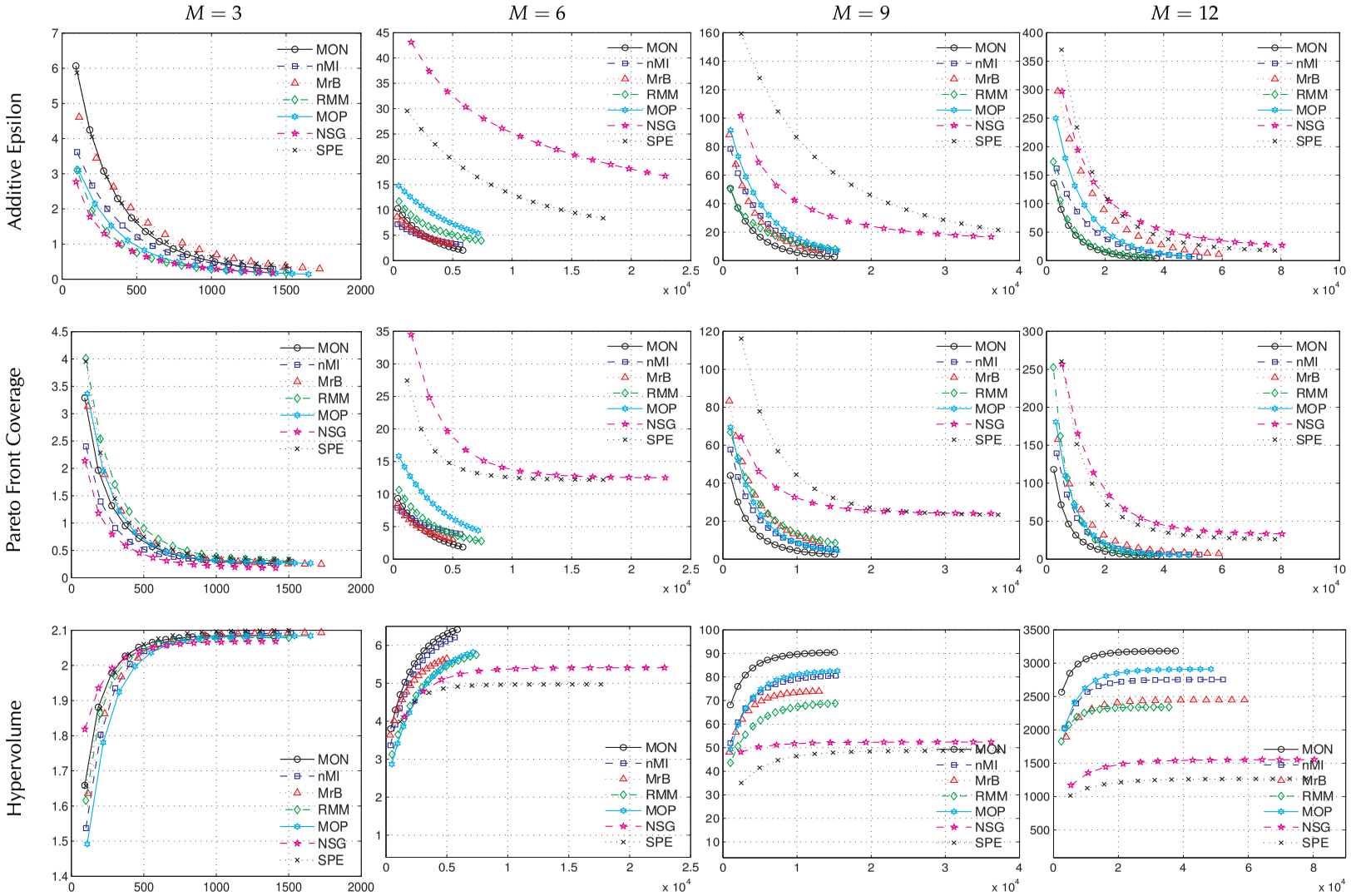


Figure D.6: Algorithms progress according to the different performance indicators while solving the WFG6 problem.

References

Ahn, C. W. (2006). *Advances in Evolutionary Algorithms. Theory, Design and Practice*. Springer. ISBN 3-540-31758-9.

Ahn, C. W., Goldberg, D. E., and Ramakrishna, R. S. (2004). Real-coded Bayesian optimization algorithm: Bringing the strength of BOA into the continuous world. In *2004 Genetic and Evolutionary Computation (GECCO 2004)*, volume 3102 of *Lecture Notes in Computer Science*, pages 840–851. Springer.

Ahn, C. W. and Ramakrishna, R. S. (2007). Multiobjective real-coded Bayesian optimization algorithm revisited: Diversity preservation. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 593–600, New York, NY, USA. ACM Press.

Ahn, C. W. and Ramakrishna, R. S. (2008). A diversity preserving selection in multiobjective evolutionary algorithms. *Applied Intelligence*, 32(3):231–248.

Auger, A. (2009a). Benchmarking the $(1+1)$–ES with one-fifth success rule on the BBOB–2009 noisy testbed. In *GECCO '09: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pages 2453–2458, New York, NY, USA. ACM.

Auger, A. (2009b). Benchmarking the $(1+1)$ evolution strategy with one-fifth success rule on the BBOB-2009 function testbed. In *GECCO '09: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pages 2447–2452, New York, NY, USA. ACM.

Auger, A. and Hansen, N. (2009). Benchmarking the $(1+1)$–CMA–ES on the BBOB–2009 noisy testbed. In *GECCO '09: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pages 2467–2472, New York, NY, USA. ACM.

Auspitz, R. and Lieben, R. (1889). *Untersuchungen uber die theorie des preises*. Duncker and Humblot, Leipzig.

Bäck, T. (1996). *Evolutionary algorithms in theory and practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, New York.

Bäck, T., Fogel, D. B., and Michalewicz, Z., editors (1997). *Handbook of Evolutionary Computation*. Institute of Physics Publishing and Oxford University Press.

Bader, J. (2010). *Hypervolume-Based Search for Multiobjective Optimization: Theory and Methods*. PhD thesis, ETH Zurich, Switzerland.

Bader, J., Deb, K., and Zitzler, E. (2010). Faster hypervolume-based search using Monte Carlo sampling. In Beckmann, M., Künzi, H. P., Fandel, G., Trockel, W., Basile, A., Drexl, A., Dawid, H., Inderfurth, K., Kürsten, W., Schittko, U., Ehrgott, M., Naujoks, B., Stewart, T. J., and Wallenius, J., editors, *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, volume 634 of *Lecture Notes in Economics and Mathematical Systems*, pages 313–326, Berlin/Heidelberg. Springer.

Bader, J. and Zitzler, E. (2008). HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. TIK Report 286, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

Bader, J. and Zitzler, E. (2011). HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76.

Baluja, S. (1994). Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, PA.

Bandyopadhyay, S., Pal, S. K., and Aruna, B. (2004). Multiobjective GAs, Quantitative Indices, and Pattern Classification. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, 34(5):2088–2099.

Basseur, M. and Zitzler, E. (2006a). Handling uncertainty in indicator–based multiobjective optimization. *International Journal of Computational Intelligence Research*, 2(3):255–272.

Basseur, M. and Zitzler, E. (2006b). A preliminary study on handling uncertainty in indicator–based multiobjective optimization. In Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J. H., Romero, J., Smith, G. D., Squillero, G., and Takagi, H., editors, *Applications of Evolutionary Computing. EvoWorkshops 2006: EvoBIO, EvoCOMNET, EvoHOT, EvoIASP, EvoINTERACTION, EvoMUSART, and EvoSTOC*, volume 3907 of *Lecture Notes in Computer Science*, pages 727–739, Budapest, Hungary. Springer.

Bellman, R. E. (1961). *Adaptive Control Processes*. Princeton University Press, Princeton, NJ.

Benjamini, Y. (1988). Opening the box of a boxplot. *The American Statistician*, 42(4):257–262.

Bernoulli, J. (1713). *Ars Conjectandi: Usum & Applicationem Praecedentis Doctrinae in Civilibus, Moralibus & Oeconomicis*, chapter Chapter 4. Self-published.

Besada, J. A., García, J., De Miguel, G., Berlanga, A., Molina, J. M., and Casar, J. R. (2005). Design of IMM filter for radar tracking using evolution strategies. *IEEE Transactions on Aerospace and Electronic Systems*, 41(3):1109–1122.

Beume, N. (2009). S–metric calculation by considering dominated hypervolume as Klee's measure problem. *Evolutionary Computation*, 17(4):477–492. PMID: 19916778.

Beume, N., Laumanns, M., and Rudolph, G. (2010). Convergence rates of $(1+1)$ evolutionary multiobjective algorithms. In *Proc. 11th Int'l. Conf. Parallel Problem Solving from*

Nature — PPSN XI, Part I, volume 5467 of *Lecture Notes in Computer Science*, pages 597–606. Springer.

Beume, N., Naujoks, B., and Emmerich, M. (2007). SMS–EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669.

Beume, N. and Rudolph, G. (2006). Faster S–metric calculation by considering dominated hypervolume as Klee's measure problem. In Kovalerchuk, B., editor, *Proceedings of the Second IASTED International Conference on Computational Intelligence*, pages 233–238. IASTED/ACTA Press.

Beyer, H.-G. and Schwefel, H.-P. (2002). Evolution strategies — A comprehensive introduction. *Natural Computing: an international journal*, 1(1):3–52.

Bishop, C. (2006). *Pattern Recognition and Machine Learning*, chapter Graphical Models, pages 359–422. Springer, New York.

Bleuler, S., Laumanns, M., Thiele, L., and Zitzler, E. (2003). PISA—A Platform and Programming Language Independent Interface for Search Algorithms. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 494–508, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Bosman, P. A. N. (2003). *Design and Application of Iterated Density-Estimation Evolutionary Algorithms*. PhD thesis, Institute of Information and Computing Sciences, Universiteit Utrecht, Utrecht, The Netherlands.

Bosman, P. A. N. and Thierens, D. (2002). Multi-objective optimization with diversity preserving mixture-based iterated density estimation evolutionary algorithms. *International Journal of Approximate Reasoning*, 31(3):259–289.

Bosman, P. A. N. and Thierens, D. (2003). The Balance Between Proximity and Diversity in Multiobjective Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 7(2):174–188.

Bosman, P. A. N. and Thierens, D. (2005). The naïve MIDEA: A baseline multi–objective EA. In Coello Coello, C. A., Hernández Aguirre, A., and Zitzler, E., editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 428–442, Guanajuato, México. Springer. Lecture Notes in Computer Science Vol. 3410.

Box, G. E. P. and Muller, M. E. (1958). A note on the generation of random normal deviates. *Annals of Mathematical Statistics*, 29:610–611.

Brandte, O. and Malinchik, S. (2004). A Broad and Narrow Approach to Interactive Evolutionary Design - An Aircraft Design Example. In et al., K. D., editor, *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part II*, pages 883–895, Seattle, Washington, USA. Springer–Verlag, Lecture Notes in Computer Science Vol. 3103.

Branke, J., Lode, C., and Shapiro, J. L. (2007). Addressing sampling errors and diversity loss in umda. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation — GECCO '07*, pages 508–515. ACM Press.

Branke, J., Miettinen, K., Deb, K., and Słowiński, R., editors (2008). *Multiobjective Optimization*, volume 5252 of *Lecture Notes in Computer Science*. Springer–Verlag, Berlin/Heidelberg.

Bringmann, K. and Friedrich, T. (2009). Approximating the least hypervolume contributor: NP–hard in general, but fast in practice. In *Evolutionary Multi-Criterion Optimization*, volume i, pages 6–20. Springer.

Bringmann, K. and Friedrich, T. (2010). Approximating the volume of unions and intersections of high–dimensional geometric objects. *Computational Geometry*, 43(6-7):601–610.

Brockhoff, D., Saxena, D. K., Deb, K., and Zitzler, E. (2008). On handling a large number of objectives a posteriori and during optimization. In Knowles, J., Corne, D., and Deb, K., editors, *Multi–Objective Problem Solving from Nature: From Concepts to Applications*, Natural Computing Series, pages 377–403. Springer.

Brockhoff, D. and Zitzler, E. (2007a). Dimensionality reduction in multiobjective optimization: The minimum objective subset problem. In Waldmann, K. H. and Stocker, U. M., editors, *Operations Research Proceedings 2006*, pages 423–429. Springer.

Brockhoff, D. and Zitzler, E. (2007b). Improving hypervolume–based multiobjective evolutionary algorithms by using objective reduction methods. In *IEEE Congress on Evolutionary Computation (CEC 2007)*, pages 2086–2093. IEEE Press.

Bui, L. T., Wesolkowski, S., Bender, A., Abbass, H., and Barlow, M. (2009). A dominance–based stability measure for multi–objective evolutionary algorithms. In *2009 IEEE Congress on Evolutionary Computation (CEC '09)*, pages 749–756, Piscataway, New Jersey. IEEE Press.

Chambers, J., Cleveland, W., Kleiner, B., and Tukey, P. (1983). *Graphical Methods for Data Analysis*. Wadsworth, Belmont.

Chernoff, H. and E. L., L. (1954). The use of maximum likelihood estimates in χ^2 tests for goodness-of-fit. *The Annals of Mathematical Statistics*, 25:579–586.

Coello Coello, C. A. (2000). An Updated Survey of GA-Based Multiobjective Optimization Techniques. *ACM Computing Surveys*, 32(2):109–143.

Coello Coello, C. A. (2004). Recent trends in evolutionary multiobjective optimization. In Abraham, A., Jain, L., and Goldberg, R., editors, *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Advanced Information and Knowledge Processing, pages 7–32. Springer–Verlag.

Coello Coello, C. A. (2006a). 20 years of evolutionary multi–objective optimization: What has been done and what remains to be done. In Yen, G. Y. and Fogel, D. B., editors, *Computational Intelligence: Principles and Practice*, chapter 4, pages 73–88. IEEE Computational Intelligence Society, Vancouver, Canada.

Coello Coello, C. A. (2006b). Evolutionary multiobjective optimization: A historical view of the field. *IEEE Computational Intelligence Magazine*, 1(1):28–36.

Coello Coello, C. A., Lamont, G. B., and Van Veldhuizen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and Evolutionary Computation. Springer, New York, second edition.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. John Wiley & Sons, New York, 3rd edition.

Cooper, G. (1990). Computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405.

Cooper, G. and Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347.

Corne, D. W. (2008). Single objective = past, multiobjective = present, ??? = future. In Michalewicz, Z., editor, *2008 IEEE Conference on Evolutionary Computation (CEC), part of 2008 IEEE World Congress on Computational Intelligence (WCCI 2008)*, Piscataway, New Jersey. IEEE Press.

Corne, D. W., Jerram, N. R., Knowles, J. D., and Oates, M. J. (2001). PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. In Spector, L., Goodman, E. D., Wu, A., Langdon, W., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. H., and Burke, E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 283–290, San Francisco, California. Morgan Kaufmann Publishers.

Corne, D. W. and Knowles, J. D. (2003). No Free Lunch and Free Leftovers Theorems for Multiobjective Optimisation Problems. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 327–341, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Corne, D. W., Knowles, J. D., and Oates, M. J. (2000). The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J. J., and Schwefel, H.-P., editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 839–848, Paris, France. Springer. Lecture Notes in Computer Science No. 1917.

Correa, E. S. and Shapiro, J. L. (2006). Model complexity vs. performance in the Bayesian optimization algorithm. In *Parallel Problem Solving from Nature-PPSN IX*, page 998–1007. Springer.

Costa, M. and Minisci, E. (2003). MOPED: A Multi-objective Parzen-Based Estimation of Distribution Algorithm for Continuous Problems. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 282–294, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Costa, M., Minisci, E., and Pasero, E. (2003). An hybrid neural/genetic approach to continuous multi–objective optimization problems. In Apolloni, B., Marinaro, M., and Tagliaferri, R., editors, *Italian Workshop on Neural Neural Nets (WIRN)*, volume 2859 of *Lecture Notes in Computer Science*, pages 61–69. Springer.

Cubo, Ó., Robles, V., Segovia, J., and Ruiz, E. M. (2005). Using genetic algorithms to improve accuracy of economical indexes prediction. In Famili, A. F., Kok, J. N., Peña, J. M., Siebes, A., and Feelders, A. J., editors, *Advances in Intelligent Data Analysis VI, 6th International Symposium on Intelligent Data Analysis (IDA 2005)*, volume 3646 of *Lecture Notes in Computer Science*. Springer.

Dagum, P. and Luby, M. (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153.

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection, or The Preservation of Favoured Races in the Struggle for Life*. John Murray, London.

Dawkins, R. (1976). *The Selfish Gene*. Oxford University Press, Oxford, UK.

De Jong, K. A. (2006). *Evolutionary Computation: A Unified Approach*. MIT Press, Cambridge, MA.

Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK. ISBN 0-471-87339-X.

Deb, K. and Jain, S. (2002). Running performance metrics for evolutionary multi-objective optimization. In *Simulated Evolution and Learning (SEAL)*, pages 13–20.

Deb, K., Miettinen, K., and Chaudhuri, S. (2010). Toward an estimation of nadir objective vector using a hybrid of evolutionary and local search approaches. *Trans. Evol. Comp*, 14:821–841.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Deb, K. and Saxena, D. K. (2005). On finding Pareto–optimal solutions through dimensionality reduction for certain large–dimensional multi–objective optimization problems. Technical Report 2005011, KanGAL.

Deb, K. and Saxena, D. K. (2006). Searching for Pareto–optimal solutions through dimensionality reduction for certain large–dimensional multi–objective optimization problems. In *2006 IEEE Conference on Evolutionary Computation (CEC'2006)*, pages 3352–3360, Piscataway, New Jersey. IEEE Press.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2004). Scalable test problems for evolutionary multiobjective optimization. In Abraham, A., Jain, L., and Goldberg, R., editors, *Evolutionary Multiobjective Optimization: Theoretical Advances and Applications*, Advanced Information and Knowledge Processing, pages 105–145. Springer Verlag.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B(39):1–38.

Deolalikar, V. (2010). P≠NP. Technical report, Hewlett Packard Research Labs, Palo Alto, CA, USA.

Edgeworth, F. Y. (1881). *Mathematical Psychics: An essay on the application of mathematics to moral sciences*. C. Keagan Paul & Co.

Ehrgott, M. (2005). *Multicriteria Optimization*, volume 491 of *Lecture Notes in Economics and Mathematical Systems*. Springer.

Eichfelder, G. (2008). *Adaptive scalarization methods in multiobjective optimization*. Vector Optimization. Springer-Verlag, Berlin/Heidelberg.

Engelbrecht, A. F. (2006). *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons, New York.

Etxeberria, R. and Larrañaga, P. (1999). Global optimization using Bayesian networks. In Ochoa, A., Soto, M. R., and Santana, R., editors, *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*, pages 151–173, Habana, Cuba.

Fleischer, M. (2003). The measure of Pareto optima. Applications to multi-objective metaheuristics. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 519–533, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Flentge, F. (2006). Locally weighted interpolating growing neural gas. *IEEE Transactions on Neural Networks*, 17(6):1382–1393.

Fogel, D. B. (2006). *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press Series on Computational Intelligence. Wiley — IEEE Press, New York.

Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, New York.

Fonseca, C. and Fleming, P. (1993a). Multiobjective Genetic Algorithms. In *IEE Colloquium on Genetic Algorithms for Control Systems Engineering*, pages 6/1–6/5. IEE.

Fonseca, C. M. and Fleming, P. J. (1993b). Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Forrest, S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California. University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers.

Fonseca, C. M., Paquete, L., and López-Ibánez, M. (2006). An improved dimension–sweep algorithm for the hypervolume indicator. In *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, pages 1157–1163.

Fritzke, B. (1994). Fast learning with incremental RBF networks. *Neural Processing Letters*, 1:2–5.

Fritzke, B. (1995). A growing neural gas network learns topologies. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, *Advances in Neural Information Processing Systems*, volume 7, pages 625–632. MIT Press, Cambridge, MA.

Fritzke, B. (1997). Incremental neuro–fuzzy systems. In *Application of Soft Computing, SPIE International Symposium on Optical Science, Engineering and Instrumentation*, San Diego, CA.

García, J., Berlanga, A., and Molina, J. M. (2009). Effective evolutionary algorithms for many–specifications attainment: Application to air traffic control tracking filters. *IEEE Transactions on Evolutionary Computation*, 13(1):151–168.

Geoffrion, A. (1968). Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications*, 22:618–630.

Goel, T. and Stander, N. (2010). A non-dominance-based online stopping criterion for multi–objective evolutionary algorithms. *International Journal for Numerical Methods in Engineering*.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison–Wesley, Reading, MA.

Gosset a.k.a. Student, W. S. (1908). The probable error of a mean. *Biometrika*, 6(1):1–25.

Grossberg, S. (1982). *Studies of Mind and Brain: Neural Principles of Learning, Perception, Development, Cognition, and Motor Control*. Reidel, Boston.

Grünwald, P. D. (2007). *The Minimum Description Length Principle (Adaptive Computation and Machine Learning)*. The MIT Press.

Guerrero, J. L., García, J., Martí, L., Molina, J. M., and Berlanga, A. (2009). A stopping criterion based on Kalman estimation techniques with several progress indicators. In Raidl, G., Alba, E., Bacardit, J., Bates Congdon, C., Beyer, H.-G., Birattari, M., Blum, C., Bosman, P. A. N., Corne, D., Cotta, C., Di Penta, M., Doerr, B., Drechsler, R., Ebner, M., Grahl, J., Jansen, T., Knowles, J., Lenaerts, T., Middendorf, M., Miller, J. F., O'Neill, M., Poli, R., Squillero, G., Stanley, K., Stützle, T., and van Hemert, J., editors, *GECCO 2009: Proceedings 11th Annual Conference on Genetic and Evolutionary Computation*, pages 587–594, New York, NY, USA. ACM Press.

Guerrero, J. L., Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010). Introducing a robust and efficient stopping criterion for MOEAs. In *2010 IEEE Conference on Evolutionary Computation (CEC), part of 2010 IEEE World Congress on Computational Intelligence (WCCI 2010)*, Piscataway, New Jersey. IEEE Press.

Hanne, T. (1999). On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research*, 117(3):553–564.

Hansen, N., Muller, S., and Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA–ES). *Evolutionary Computation*, 11(1):1–18.

Hansen, N. and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195.

Hartigan, J. A. (1975). *Clustering Algorithms*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, New York.

Heckerman, D. and Wellman, M. P. (1995). Bayesian networks. *Communications of the ACM*, 38(3):27–30.

Hernandez, G., Wilder, K., Nino, F., and Garcia, J. (2005). Towards a self-stopping evolutionary algorithm using coupling from the past. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 615–620, New York, NY, USA. ACM.

Hofstadter, D. R. (1985). *Metamagical Themas: Questing for the Essence of Mind and Pattern*. Basic Books, New York.

Hong, Y., Kwong, S., Ren, Q., and Wang, X. (2007). Over-selection: An attempt to boost EDA under small population size. In *2007 IEEE Congress on Evolutionary Computation*, pages 1075–1082. IEEE Press.

Horn, J., Nafpliotis, N., and Goldberg, D. E. (1994). A Niched Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscataway, New Jersey. IEEE Service Center.

Hornby, G. S. and Yu, T. (2007). EC practitioners: Results of the first survey. *SIGEVOlution*, 2(1):2–8.

Huband, S., Barone, L., While, L., and Hingston, P. (2005). A scalable multi-objective test problem toolkit. In Coello Coello, C. A., Hernández Aguirre, A., and Zitzler, E., editors, *Evolutionary Multi-Criterion Optimization. Third International Conference, EMO 2005*, pages 280–295, Guanajuato, México. Springer. Lecture Notes in Computer Science Vol. 3410.

Huband, S., Hingston, P., Barone, L., and While, L. (2006a). A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

Huband, S., Hingston, P., Barone, L., and While, L. (2006b). A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

Igel, C., Hansen, N., and Roth, S. (2007). Covariance matrix adaptation for multi-objective optimization. *Evolutionary Computation*, 15(1):1–28.

Isasi, P. and Hernandez, J. C. (2004). Introduction to the applications of evolutionary computation in computer security and cryptography. *Computational Intelligence*, 20(3):445–449.

Ishibuchi, H. and Murata, T. (1998). Multi-Objective Genetic Local Search Algorithm and Its Application to Flowshop Scheduling. *IEEE Transactions on Systems, Man and Cybernetics—Part C: Applications and Reviews*, 28(3):392–403.

Ishibuchi, H., Tsukamoto, N., and Nojima, Y. (2008). Evolutionary many–objective optimization: A short review. In *IEEE Congress on Evolutionary Computation, 2008 (CEC 2008) part of IEEE World Congress on Computational Intelligence*, pages 2419–2426, Piscataway, New Jersey. IEEE Press.

Ishibuchi, H., Tsukamoto, N., and Nojima, Y. (2010). Diversity improvement by non-geometric binary crossover in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 14(6):985–998.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transaction of the ASME — Journal of Basic Engineering*, 82:35–45.

Kambhatla, N. and Leen, T. K. (1997). Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516.

Karush, W. (1939). Minima of functions of several variables with inequalities as side constraints. Master's thesis, Department of Mathematics, University of Chicago, Chicago, IL, USA.

Khan, N. (2003). Bayesian Optimization Algorithms for Multiobjective and Hierarchically Difficult Problems. Master's thesis, Graduate College of the University of Illinois at Urbana-Champaign, Urbana, Illinois, USA.

Khan, N., Goldberg, D. E., and Pelikan, M. (2002). Multi-Objective Bayesian Optimization Algorithm. In Langdon, W., Cantú-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M., Schultz, A., Miller, J., Burke, E., and Jonoska, N., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002)*, page 684, San Francisco, California. Morgan Kaufmann Publishers.

Khare, V. (2002). Performance Scaling of Multi-Objective Evolutionary Algorithms. Master's thesis, School of Computer Science, The University of Birmingham, Edgbaston, Birmingan, UK.

Khare, V., Yao, X., and Deb, K. (2003). Performance Scaling of Multi-objective Evolutionary Algorithms. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 376–390, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Kibler, D., Aha, D. W., and Albert, M. K. (1989). Instance–based prediction of real–valued attributes. *Computational Intelligence*, 5(2):51–57.

Kindermann, R. and Snell, J. L. (1980). *Markov Random Fields and Their Applications*. Contemporary Mathematics. American Mathematical Society, Providence, RI.

Knowles, J. and Corne, D. (2007). Quantifying the effects of objective space dimension in evolutionary multiobjective optimization. In Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., and Murata, T., editors, *Proceedings 4th International Conference Evolutionary Multi–Criterion Optimization (EMO 2007)*, pages 757–771. Springer, Berlin/Heidelberg.

Knowles, J., Thiele, L., and Zitzler, E. (2006a). A tutorial on the performance assessment of stochastic multiobjective optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich.

Knowles, J., Thiele, L., and Zitzler, E. (2006b). A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland. revised version.

Knowles, J. D. (2002). *Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization*. PhD thesis, The University of Reading, Department of Computer Science, Reading, UK.

Knowles, J. D. and Corne, D. W. (1999). The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation*, pages 98–105, Washington, D.C. IEEE Service Center.

Knowles, J. D. and Corne, D. W. (2000). Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172.

Korhonen, P. (2005). Interactive methods. In Figueira, J., Greco, S., and Ehrogott, M., editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, volume 78 of *International Series in Operations Research & Management Science*, chapter 16, pages 641–661. Springer, Berlin/Heidelberg.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, MA.

Kruskal, W. H. and Wallis, W. A. (1952). Use of ranks in one-criterion analysis of variance. *Journal of the American Statistical Association*, 47:583–621.

Kuhn, H. W. and Tucker, A. W. (1951). Nonlinear programming. In Neyman, J., editor, *Proceedings of Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, Berkeley, LA, USA. University of California Press.

Lamarck, J.-B. (1809). *Philosophie zoologique, ou Exposition des considérations relatives à l'histoire naturelle des animaux*. . . . Dentu, Paris, France.

Larrañaga, P. and Lozano, J. A., editors (2002). *Estimation of Distribution Algorithms. A new tool for Evolutionary Computation*. Genetic Algorithms and Evolutionary Computation. Kluwer Academic Publishers, Boston/Dordrecht/London.

Laumanns, M. and Ocenasek, J. (2002). Bayesian Optimization Algorithms for Multi-objective Optimization. In Merelo Guervós, J. J., Adamidis, P., Beyer, H.-G., nas, J.-L. F.-V., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature—PPSN VII*, pages 298–307, Granada, Spain. Springer–Verlag. Lecture Notes in Computer Science No. 2439.

Levon, J. (2004). *OProfile manual*. Victoria University of Manchester.

Lozano, J. A., Larrañaga, P., Inza, I., and Bengoetxea, E., editors (2006). *Towards a New Evolutionary Computation: Advances on Estimation of Distribution Algorithms*. Springer–Verlag.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical*, volume 1, pages 281–297.

Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18:50–60.

Martí, L., García, J., Berlanga, A., Coello Coello, C. A., and Molina, J. M. (2010a). On current model-building methods for multi-objective estimation of distribution algorithms: Shortcommings and directions for improvement. Technical Report GIAA2010E001, Grupo de Inteligencia Artificial Aplicada, Universidad Carlos III de Madrid, Colmenarejo, Spain.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2007). A cumulative evidential stopping criterion for multiobjective optimization evolutionary algorithms. In Thierens, D., Deb, K., Pelikan, M., Beyer, H.-G., Doerr, B., Poli, R., and Bittari, M., editors, *GECCO '09: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, page 911, New York. ACM Press.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2008a). Introducing MONEDA: Scalable multiobjective optimization with a neural estimation of distribution algorithm. In Keizer, M., Antoniol, G., Congdon, C., Deb, K., Doerr, B., Hansen, N., Holmes, J., Hornby, G., Howard, D., Kennedy, J., Kumar, S., Lobo, F., Miller, J., Moore, J., Neumann, F., Pelikan, M., Pollack, J., Sastry, K., Stanley, K., Stoica, A., Talbi, E. G., and Wegener, I., editors, *GECCO '08: 10th Annual Conference on Genetic and Evolutionary Computation*, pages 689–696, New York, NY, USA. ACM Press. EMO Track "Best Paper" Nominee.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2008b). Model-building algorithms for multiobjective EDAs: Directions for improvement. In *2008 IEEE Conference on Evolutionary Computation (CEC), part of 2008 IEEE World Congress on Computational Intelligence (WCCI 2008)*, pages 2848–2855, Piscataway, New Jersey. IEEE Press.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2008c). Scalable continuous multiobjective optimization with a neural network–based estimation of distribution algorithm. In Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G. A., Drechsler, R., Ekárt, A., Esparcia-Alcázar, A. I., Farooq, M., Fink, A., McCormack, J., O'Neill, M., Romero, J., Rothlauf, F., Squillero, G., Uyar, A. c., and Yang, S., editors, *Applications of Evolutionary Computing*, volume 4974 of *Lecture Notes in Computer Science*, pages 535–544, Heidelberg. Springer.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009a). An approach to stopping criteria for multi–objective optimization evolutionary algorithms: The MGBM criterion. In *2009 IEEE Conference on Evolutionary Computation (CEC 2009)*, pages 1263–1270, Piscataway, New Jersey. IEEE Press.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009b). On the computational properties of the multi–objective neural estimation of distribution algorithms. In Krasnogor, N., Melián-Batista, B., Moreno-Pérez, J. A., Moreno-Vega, J. M., and Pelta, D., editors, *Nature Inspired Cooperative Strategies for Optimization (NICSO 2008)*, volume 236 of *Studies in Computational Intelligence*, pages 239–251, Berlin/Heidelberg. Springer.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009c). On the model–building issue of multi–objective estimation of distribution algorithms. In Corchado, E., Wu, X., Oja, E., Herrero, A., and Baruque, B., editors, *4th International Conference on Hybrid Artificial Intelligence (HAIS'09)*, volume 5572 of *Lecture Notes in Artificial Intelligence*, pages 293–300, Berlin/Heidelberg. Springer.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2009d). Solving complex high-dimensional problems with the multi-objective neural estimation of distribution algorithm. In Raidl, G., Alba, E., Bacardit, J., Bates Congdon, C., Beyer, H.-G., Birattari, M., Blum, C., Bosman, P. A. N., Corne, D., Cotta, C., Di Penta, M., Doerr, B., Drechsler, R., Ebner, M., Grahl, J., Jansen, T., Knowles, J., Lenaerts, T., Middendorf, M., Miller, J. F., O'Neill, M., Poli, R., Squillero, G., Stanley, K., Stützle, T., and van Hemert, J., editors, *GECCO 2009: 11th Annual Conference on Genetic and Evolutionary Computation*, pages 619–626, New York, NY, USA. ACM Press.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010b). Advancing model-building for many-objective optimization estimation of distribution algorithms. In Chio, C. D., Cagnoni, S., Cotta, C., Ebner, M., Ekárt, A., Esparcia-Alcazar, A. I., Goh, C.-K., Merelo, J. J., Neri, F., Preuß, M., Togelius, J., and Yannakakis, G. N., editors, *Applications of Evolutionary Computation*, volume 6024 of *Lecture Notes in Computer Science*, pages 512–521. Springer.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010c). MONEDA: Scalable multi–objective optimization with a neural network–based estimation of distribution algorithm. Technical Report GIAA2010E002, Grupo de Inteligencia Artificial Aplicada, Universidad Carlos III de Madrid, Colmenarejo, Spain.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010d). Moving away from error-based learning in multi-objective estimation of distribution algorithms. In Branke, J., Alba, E., Arnold, D., Bongard, J., Brabazon, A., Butz, M. V., Clune, J., Cohen, M., Deb, K., Engelbrecht, A., Krasnogor, N., Miller, J., O'Neill, M., Sastry, K., Thierens, D., Vanneschi, L., van Hemert, J., and Witt, C., editors, *GECCO '10: Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 545–546, New York, NY, USA. ACM Press.

Martí, L., García, J., Berlanga, A., and Molina, J. M. (2010e). A progress indicator for detecting success and failure in evolutionary multi–objective optimization. In *2010 IEEE Conference on Evolutionary Computation (CEC), part of 2010 IEEE World Congress on Computational Intelligence (WCCI 2010)*, Piscataway, New Jersey. IEEE Press.

Martinetz, T. M. (1993). Competitive Hebbian learning rule forms perfectly topology preserving maps. In *International Conference on Artificial Neural Networks (ICANN'93)*, pages 427–434, Amsterdam. Springer–Verlag.

Martinetz, T. M., Berkovich, S. G., and Shulten, K. J. (1993). Neural–gas network for vector quantization and its application to time–series prediction. *IEEE Transactions on Neural Networks*, 4:558–560.

Massey, F. J. (1951). The Kolmogorov–Smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78.

Maybeck, P. S. (1979). *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Academic Press.

Mendes, S., Molina, G., Vega-Rodriguez, M., Gomez-Pulido, J., Saez, Y., Miranda, G., Segura, C., Alba, E., Isasi, P., Leon, C., and Sanchez-Perez, J. (2009). Benchmarking a wide spectrum of metaheuristic techniques for the radio network design problem. *IEEE Transactions on Evolutionary Computation*, 13(5):1133–1150.

Mersmann, O., Trautmann, H., Naujoks, B., and Weihs, C. (2010). On the distribution of EMOA hypervolumes. In Blum, C. and Battiti, R., editors, *LION*, volume 6073 of *Lecture Notes in Computer Science*, pages 333–337. Springer.

Michalski, R. S. (2000). Learnable evolution model: Evolutionary processes guided by machine learning. *Machine Learning*, 38:9–40.

Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*, volume 12 of *International Series in Operations Research & Management Science*. Kluwer, Norwell, MA.

Miquélez, T., Bengoetxea, E., and Larrañaga, P. (2004). Evolutionary computation based on Bayesian classifiers. *International Journal of Applied Mathematics and Computer Science*, 14(3):335–349.

Miquélez, T., Bengoetxea, E., and Larrañaga, P. (2006a). *Bayesian Classifiers in Optimization: An EDA-like Approach*, volume 242, pages 221–242. Springer.

Miquélez, T., Bengoetxea, E., and Larrañaga, P. (2006b). *Evolutionary Bayesian Classifier-based Optimization in Continuous Domains*, pages 529–536. Springer.

Mochón, A., Quintana, D., Sáez, Y., and Isasi, P. (2008). Soft computing techniques applied to finance. *Applied Intelligence*, 29:111–115.

Mühlenbein, H. and Mahnig, T. (1999). FDA – a scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, 7(4):353–376.

Mühlenbein, H. and Paaß, G. (1996). From recombination of genes to the estimation of distributions I. Binary parameters. In Voigt, H.-M., Ebeling, W., Rechenberg, I., and Schwefel, H.-P., editors, *Parallel Problem Solving from Nature - PPSN IV*, pages 178–187, Berlin. Springer Verlag. LNCS 1141.

Nakayama, H., Kaneshige, S., Takemoto, S., and Watada, Y. (1995). An application of a multi–objective programming technique to construction accuracy control of cable–stayed bridges. *European Journal of Operational Research*, 87:731–738.

Naujoks, B. and Trautmann, H. (2009). Online convergence detection for multiobjective aerodynamic applications. In Tyrrell, A. et al., editors, *Proc. Int'l. Congress on Evolutionary Computation (CEC 2009)*, pages 332–339, Piscataway, NJ. IEEE press.

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization*. Series in Operations Research. Springer.

Ocenasek, J. (2002). *Parallel Estimation of Distribution Algorithms*. PhD thesis, Faculty of Information Technology, Brno University of Technology, Brno, Czech Republic.

Ocenasek, J. and Schwarz, J. (2002). Estimation of distribution algorithm for mixed continuous–discrete optimization problems. In *2nd Euro–International Symposium on Computational Intelligence*, pages 227–232.

Papadimitriou, C. M. (1994). *Computational Complexity*. Addison–Wesley, Reading, MA, USA.

Pareto, V. (1896). *Cours D'Économie Politique*. F. Rouge, Lausanne.

Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathernatical Statistics*, 33:1065–1076.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA.

Pelikan, M. (2005a). *Hierarchical Bayesian Optimization Algorithm. Toward a New Generation of Evolutionary Algorithms*. Studies in Fuzziness and Soft Computing. Springer.

Pelikan, M. (2005b). *Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms*. Studies in Fuzziness and Soft Computing. Springer, 1 edition.

Pelikan, M. and Goldberg, D. E. (2006). Hierarchical bayesian optimization algorithm. In Pelikan, M., Sastry, K., and Cantú-Paz, E., editors, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, Studies in Computational Intelligence, pages 63–90. Springer–Verlag.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (1999a). BOA: The Bayesian optimization algorithm. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., and Smith, R. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-1999*, volume I, pages 525–532, Orlando, FL. Morgan Kaufmann Publishers, San Francisco, CA.

Pelikan, M., Goldberg, D. E., and Cantú-Paz, E. (2000). Hierarchical problem solving by the Bayesian optimization algorithm. IlliGAL Report No. 2000002, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL.

Pelikan, M., Goldberg, D. E., and Lobo, F. (1999b). A survey of optimization by building and using probabilistic models. IlliGAL Report No. 99018, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL.

Pelikan, M., Sastry, K., and Cantú-Paz, E., editors (2006a). *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*. Studies in Computational Intelligence. Springer.

Pelikan, M., Sastry, K., and Goldberg, D. E. (2005). Multiobjective hBOA, clustering, and scalability. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 663–670, New York, NY, USA. ACM Press.

Pelikan, M., Sastry, K., and Goldberg, D. E. (2006b). Multiobjective estimation of distribution algorithms. In Pelikan, M., Sastry, K., and Cantú-Paz, E., editors, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, Studies in Computational Intelligence, pages 223–248. Springer–Verlag.

Pena, J. M., Robles, V., Larrañaga, P., Herves, V., Rosales, F., and Pérez, M. S. (2004). GA-EDA: Hybrid evolutionary algorithm using genetic and estimation of distribution algorithms. In *Innovations in Applied Artificial Intelligence*, pages 361–371, Heidelberg/Berlin. Springer.

Praditwong, K. and Yao, X. (2007). How well do multi–objective evolutionary algorithms scale to large problems. In *2007 IEEE Congress on Evolutionary Computation (CEC 2007)*, pages 3959–3966, Piscataway, New Jersey. IEEE Press.

Purshouse, R. C. (2003). *On the Evolutionary Optimisation of Many Objectives*. PhD thesis, Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK.

Purshouse, R. C. and Fleming, P. J. (2003). Evolutionary Multi-Objective Optimisation: An Exploratory Analysis. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 3, pages 2066–2073, Canberra, Australia. IEEE Press.

Purshouse, R. C. and Fleming, P. J. (2007). On the evolutionary optimization of many conflicting objectives. *IEEE Transactions on Evolutionary Computation*, 11(6):770–784.

Qin, A. K. and Suganthan, P. N. (2004). Robust growing neural gas algorithm with application in cluster analysis. *Neural Networks*, 17(8–9):1135–1148.

Quinlan, J. R. (1993). Combining instance-based and model-based learning. In *Machine Learning: Proceedings of the Tenth International Conference*, pages 236–243. Morgan Kaufmann.

Rechenberg, I. (1973). *Evolutionsstrategie — Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Fromman–Holzboog, Stuttgart, Germany.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo method*. John Wiley & Sons, New York.

Rudenko, O. and Schoenauer, M. (2004). A steady performance stopping criterion for pareto–based evolutionary algorithms. In *The 6th International Multi–Objective Programming and Goal Programming Conference*, Hammamet, Tunisia.

Rudolph, G. and Agapie, A. (2000). Convergence Properties of Some Multi-Objective Evolutionary Algorithms. In *Proceedings of the 2000 Conference on Evolutionary Computation*, volume 2, pages 1010–1016, Piscataway, New Jersey. IEEE Press.

Safe, M., Carballido, J., Ponzoni, I., and Brignole, N. (2004). On stopping criteria for genetic algorithms. In *Advances in Artificial Intelligence – SBIA 2004*, pages 405–413. Springer, Berlin/Heidelberg.

Santana, P. and Correia, L. (2011). Swarm cognition on off-road autonomous robots. *Swarm Intelligence*, 5(1):45–72. Article first published online: 4 January 2011.

Santana, R. (2003). A Markov network based factorized distribution algorithm for optimization. In Lavrač, N., Gamberger, D., Blockeel, H., and Todorovski, L., editors, *Machine Learning: ECML 2003*, volume 2837 of *Lecture Notes in Artificial Intelligence*, pages 337–348. Springer, Berlin/Heidelberg/New York.

Santana, R. (2005). Estimation of distribution algorithms with Kikuchi approximations. *Evolutionary Computation*, 13(1):67–97.

Santana, R., Larrañaga, P., and Lozano, J. A. (2008). Protein folding in simplified models with estimation of distribution algorithms. *IEEE Transactions on Evolutionary Computation*, 12(4):418–438.

Sarle, W. S. (1995). Why statisticians should not FART. Technical report, SAS Institute, Cary, NC.

Schaffer, J. D. (1985). Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum.

Schervish, M. J. (1997). *Theory of Statistics*. Springer Series in Statistics. Springer, Berlin/Heidelberg, second edition.

Scholköpf, B., Smola, A., and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319.

Schütze, O., Mostaghim, S., Dellnitz, M., and Teich, J. (2003). Covering Pareto Sets by Multilevel Evolutionary Subdivision Techniques. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 118–132, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464.

Schwefel, H.-P. (1977). *Numerische Optimierung von Computer-Modellen*. Birkhäuser Verlag, Basel, Switzerland.

Shakya, S. and McCall, J. (2007). Optimization by estimation of distribution with DEUM framework based on markov random fields. *International Journal of Automation and Computing*, 4(3):262–272.

Shapiro, J. (2006). Diversity loss in general estimation of distribution algorithms. In *Parallel Problem Solving from Nature - PPSN IX*, pages 92–101.

Sheri, G. and Corne, D. W. (2008). The simplest evolution/learning hybrid: LEM with KNN. In *IEEE World Congress on Computational Intelligence*, pages 3244–3251, Hong Kong, China. IEEE Press.

Sheri, G. and Corne, D. W. (2010). Learning-assisted evolutionary search for scalable function optimization: LEM(ID3). In *IEEE World Congress on Computational Intelligence*, Barcelona, Spain. IEEE Press.

Srinivas, N. and Deb, K. (1994). Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248.

Stewart, T. J., Bandte, O., Braun, H., Chakraborti, N., Ehrgott, M., Göbelt, M., Jin, Y., Nakayama, H., Poles, S., and Di Stefano, D. (2008). Real–world applications of multiobjective optimization. In Branke, J., Miettinen, K., Deb, K., and Słowiński, R., editors, *Multiobjective Optimization*, volume 5252 of *Lecture Notes in Computer Science*, pages 285–327. Springer–Verlag, Berlin/Heidelberg.

Stewart, T. J., Janssen, R., and van Herwijnen, M. (2004). A genetic algorithm approach to multiobjective land use planning. *Computers and Operations Research*, 32:2293–2313.

Storn, R. and Price, K. (1997). Differential evolution — A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359.

Thierens, D. (2003). Convergence Time Analysis for the Multi-objective Counting Ones Problem. In Fonseca, C. M., Fleming, P. J., Zitzler, E., Deb, K., and Thiele, L., editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 355–364, Faro, Portugal. Springer. Lecture Notes in Computer Science. Volume 2632.

Thierens, D. and Bosman, P. A. N. (2001a). Multi-objective mixture-based iterated density estimation evolutionary algorithms. In Spector, L., Goodman, E., Wu, A., Langdon, W., Voigt, H., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M., and Burke, E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-2001*, pages 663–670, San Francisco, CA. Morgan Kaufmann Publishers.

Thierens, D. and Bosman, P. A. N. (2001b). Multi-Objective Optimization with Iterated Density Estimation Evolutionary Algorithms using Mixture Models. In *Proceedings of the Third International Symposium on Adaptive Systems—Evolutionary Computation and Probabilistic Graphical Models*, pages 129–136, Havana, Cuba. Institute of Cybernetics, Mathematics and Physics.

Timm, H., Borgelt, C., Doring, C., and Kruse, R. (2004). An extension to possibilistic fuzzy cluster analysis. *Fuzzy Sets and Systems*, 147(1):3–16.

Trautmann, H., Wagner, T., Naujoks, B., Preuss, M., and Mehnen, J. (2009). Statistical methods for convergence detection of multi-objective evolutionary algorithms. *Evolutionary Computation*, 17(4):493–509. PMID: 19916777.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999.

Wagner, T., Beume, N., and Naujoks, B. (2007). Pareto–, aggregation–, and indicator–based methods in many–objective optimization. In Obayashi, S., Deb, K., Poloni, C., Hiroyasu, T., and Murata, T., editors, *4th International Conference on Evolutionary Multi–Criterion Optimization (EMO 2007)*, volume 4403 of *Lecture Notes in Computer Science*, pages 742–756, Heidelberg. Springer.

Wagner, T. and Martí, L. (2010). Taxonomy-based Matlab framework for online stopping criteria. Online (http://www.giaa.inf.uc3m.es/miembros/lmarti/stopping).

Wagner, T. and Trautmann, H. (2010). Online convergence detection for evolutionary multi-objective algorithms revisited. In Fogel, G. and Ishibuchi, H., editors, *Proc. Int'l. Congress on Evolutionary Computation (CEC 2010)*, pages 3554–3561, Piscataway, NJ. IEEE press.

Wagner, T., Trautmann, H., and Martí, L. (2011). A taxonomy of online stopping criteria for multi-objective evolutionary algorithms. In Takahashi, R. H. C., Deb, K., Wanner, E. F., and Greco, S., editors, *6th International Conference on Evolutionary Multi-Criterion Optimization (EMO 2011)*, volume 6576, pages 16–30, Berlin/Heidelberg. Springer. 10.1007/978-3-642-19893-9_2.

Wagner, T., Trautmann, H., and Naujoks, B. (2009). OCD: Online convergence detection for evolutionary multi–objective algorithms based on statistical testing. In Ehrgott, M., Fonseca, C. M., Gandibleux, X., Hao, J.-K., and Sevaux, M., editors, *5th International Conference on Evolutionary Multi–Criterion Optimization (EMO 2009)*, volume 5467 of *Lecture Notes in Computer Science*, pages 198–215. Springer.

Wallin, D. and Ryan, C. (2007). On the diversity of diversity. In *2007 IEEE Congress on Evolutionary Computation*, pages 95–102. IEEE.

While, L., Bradstreet, L., Barone, L., and Hingston, P. (2005). Heuristics for Optimising the Calculation of Hypervolume for Multi-Objective Optimization Problems. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 3, pages 2225–2232, Edinburgh, Scotland. IEEE Service Center.

While, L., Hingston, P., Barone, L., and Huband, S. (2006). A faster algorithm for calculating hypervolume. *IEEE Transactions on Evolutionary Computation*, 10(1):29–38.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1:80–83.

Williamson, J. R. (1996). Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. *Neural Networks*, 9:881–897.

Wilson, E. B. and Hilferty, M. M. (1931). The distribution of chi-square. *Proceedings of the National Academy of Sciences*, 17:684–688.

Xu, R. and Wunsch II, D. (2008). *Clustering*. IEEE Press Series on Computational Intelligence. Wiley — IEEE Press, New York, illustrated edition.

Yuan, B. and Gallagher, M. (2005). On the importance of diversity maintenance in estimation of distribution algorithms. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 719–726, New York, NY, USA. ACM Press.

Zhang, Q., Zhou, A., and Jin, Y. (2008). RM–MEDA: A regularity model–based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 12(1):41–63.

Zhou, A., Zhang, Q., Jin, Y., Tsang, E., and Okabe, T. (2005). A Model-Based Evolutionary Algorithm for Bi-objective Optimization. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 3, pages 2568–2575, Edinburgh, Scotland. IEEE Service Center.

Zielinski, K. and Laur, R. (2007). Stopping criteria for a constrained single–objective particle swarm optimization algorithm. *Informatica*, 31:51–59.

Zielinski, K., Peters, D., and Laur, R. (2005). Stopping criteria for single–objective optimization. In *Third International Conference on Computational Intelligence, Robotics and Autonomous Systems*.

Zitzler, E., Brockhoff, D., and Thiele, L. (2007). The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In Obayashi, S. et al., editors, *Conference on Evolutionary Multi–Criterion Optimization (EMO 2007)*, volume 4403 of *LNCS*, pages 862–876, Berlin. Springer.

Zitzler, E., Deb, K., and Thiele, L. (1999). Comparison of Multiobjective Evolutionary Algorithms on Test Functions of Different Difficulty. In Wu, A. S., editor, *Proceedings of the 1999 Genetic and Evolutionary Computation Conference. Workshop Program*, pages 121–122, Orlando, Florida.

Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195.

Zitzler, E., Knowles, J., and Thiele, L. (2008). Quality assessment of pareto set approximations. In Branke, J., Miettinen, K., Deb, K., and Słowiński, R., editors, *Multiobjective Optimization*, volume 5252 of *Lecture Notes in Computer Science*, pages 373–404. Springer–Verlag, Berlin/Heidelberg.

Zitzler, E. and Künzli, S. (2004). Indicator-based selection in multiobjective search. In Yao, X., editor, *Parallel Problem Solving from Nature — PPSN VIII*, volume 3242 of *Lecture Notes in Computer Science*, pages 832–842, Berlin/Heidelberg. Springer–Verlag.

Zitzler, E., Laumanns, M., and Thiele, L. (2002a). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In Giannakoglou, K., Tsahalis, D., Periaux, J., Papailou, P., and Fogarty, T., editors, *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece.

Zitzler, E., Laumanns, M., Thiele, L., Fonseca, C. M., and Grunert da Fonseca, V. (2002b). Why Quality Assessment of Multiobjective Optimizers Is Difficult. In Langdon, W. B., Cantú-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M., Schultz, A., Miller, J., Burke, E., and Jonoska, N., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002)*, pages 666–673, San Francisco, California. Morgan Kaufmann Publishers.

Zitzler, E. and Thiele, L. (1998). An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich, Switzerland.

Zitzler, E., Thiele, L., and Bader, J. (2010). On set-based multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 14(1):58–79.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and Grunert da Fonseca, V. (2003). Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132.

Index

Notation Summary

$\preccurlyeq_{\epsilon+}$	Additive epsilon dominance relation
$\preccurlyeq_{\epsilon+}$	Multiplicative epsilon dominance relation
α	Selection percentile
δ_{G}	Error redistribution rate for worst neighbor of worst node
δ_{I}	Error redistribution rate for worst node
$\prec$	Pareto dominance relation
ϵ_{best}	Best-matching node learning rate
ϵ_{vic}	Neighbors of best-matching node learning rate
$\hat{\mathcal{P}}_t$	Model-building dataset
ω	Substitution percentile
ρ	MB-GNG training stopping threshold
$\mathcal{D}$	Decision set
$\mathcal{O}$	Objective set
$\mathcal{O}^*$	Pareto-optimal front
$\mathcal{PF}^*$	Local non-dominated front
$\mathcal{P}^*$	Local non-dominated set
$\mathcal{P}_t^*$	Non-dominated subset of a $\mathcal{P}_t$
$\mathcal{P}_t$	Population at iteration t
$\mathcal{S}$	Feasible set
$\mathcal{S}^*$	Pareto-optimal set
$\prec\!\!\prec$	Strict Pareto dominance relation
$\preccurlyeq$	Weak Pareto dominance relation
ξ_i	Accumulated error of node i
$f_i(\cdot)$	Objective function
I_{cov}	Pareto-optimal front coverage indicator

I_{hyp} Hypervolume indicator

I_{mdr} Mutual domination rate indicator (MDR)

$I_{\epsilon+}$ Additive epsilon indicator

M Dimension of objective set (number of objective functions)

n Dimension of decision set (number of variables)

N_{max} Maximum number of MB-GNG nodes

n_{pop} Number of individuals of a given population